

MA-004-43-1  
F.S.D .....N° D'ordre .....

Université Saad DAHLAB de Blida



Faculté des Sciences

Département d'Informatique

Mémoire présenter par :

M<sup>elle</sup> KRELIFA Halla

M<sup>elle</sup> ZENILE Amina

En vue d'obtenir le diplôme de Master

**Domaine:** MI

**Filière:** Informatique

**Spécialité:** Informatique

**Option:** Ingénierie de logiciel

Sujet :

**Intégration des sources de métadonnées hétérogènes :  
XML, RDF et RuleML par l'approche médiateur**

Soutenue le:

, devant le jury composé de :

M. FERFERA  
M. KERHICH  
M. Sid oumou  
M<sup>me</sup> M. FAREH

Président  
Rapporteur  
Examineur  
Promotrice

MA-004-43-1

## **REMERCIEMENT**

*Avant toute chose, nous tenons à remercier le Dieu le tout puissant de nous avoir donné la volonté, la patience et le courage de poursuivre et d'achever ce travail dans les bonnes conditions.*

*Nos remerciements également à notre promotrice Mme FAREH, pour nous avoir encadré, diriger et conseiller tout au long de ces mois de travail ainsi pour son aide précieuse lors de la correction de ce mémoire ;*

*Nous espérons que nos professeurs de l'USDB trouveront ici, l'expression de notre gratitude, qu'ils soient vivement remerciés des efforts considérables qu'ils ont fournis pour notre formation ;*

*Nos remerciements s'adressent aussi à Affaf-Hana et halima pour leurs encouragements et leur gentillesse.*

*Nous n'oublierons pas le soutien constant et les encouragements de notre grande famille.*

*En fin, on tient à remercier les membres de jury qui vont faire l'honneur d'apprécier notre travail.*

*Merci*

# *Dédicace*

*Je dédie ce modeste travail à mes très chers parents qui m'ont toujours soutenu pendant toute ma vie, je leur souhaite le bonheur et la bonne santé.*

*A toute ma grande famille ; mes Sœurs: **Nassima, Sihem, Ilhem et mon frère Mohamed,***

*A toute ma famille oncles tantes cousins cousines ;*

*A ma collègue de binôme Halla et sa famille;*

*Et a tout mes amis, et en particulier à : **Amina, Ratiba, Merieme, Manel, Habiba, Noura.***

*Amina*

# *Dédicace*

*Je dédie ce modeste travail à mes très chers parents qui  
m'ont toujours soutenu pendant toute ma vie.*

*A toute ma famille ; mon frère et mes Sœurs  
Et surtout ma grande mère;*

*A ma collègue de binôme Amina, et sa grande famille;*

*A tous mes collègues.*

*A toute personne porte des sentiments pour moi ;*

*Merci à tous ceux qui m'ont aidé de près ou de loin.*

*Et a tout mes amis*

*Halla*

## **Résumé**

La masse énorme d'informations disponibles sur des sources de données distribuées et hétérogènes, nécessite une politique ou un plan de recherche, de sélection et d'analyse, de plus en plus performants, pour permettre à l'utilisateur de localiser et extraire précisément l'information désirée d'une manière simple et efficace.

Notre travail consiste à proposer une architecture de médiation des sources de métadonnées hétérogènes représentées par le XML, RDF et RuleML, assurant à l'utilisateur la transparence de l'hétérogénéité. Ceci en intégrant des données de structures et de natures fondamentalement différentes et en permettant la décomposition d'une requête faisant intervenir plusieurs sources en des requêtes spécifiques à ces sources, puis savoir recomposer le résultat.

**Mots Clés:** Métadonnée, Intégration, Médiateur, XML, Traitement de requêtes.

## **Abstract**

The enormous mass of information available on heterogeneous data sources, requires a policy or a plan of retrieval, selection and analysis, increasingly powerful, to make it possible to the user to locate and to precisely extract desired information in a simple and effective way.

Our work consists in proposing mediation architecture of heterogeneous sources of metadata represented by XML, RDF and RuleML ensuring to the user the transparency of the heterogeneity. This by integrating data of structures and natures basically different and by allowing the decomposition from a query utilizing several sources in specific queries to these sources, then to know to recombine the result, while carrying out the plan of lower cost and by taking account of the sources capacities. Moreover hold account owing to the fact that metadata sources are semi-structured.

**Key Words:** Metadata, Integration, Mediator, XML, Query processing.

## ملخص

إن الكم الهائل من المعلومات المتاحة عن مصادر البيانات الموزعة وغير المتجانسة تتطلب سياسة أو خطة لاختيار واسترجاع وتحليل على نحو متزايد، لتجعل من الممكن للمستخدم تحديد مكان واستخراج المعلومات المطلوبة بدقة وبطريقة بسيطة وفعالة.

عملنا هذا يتمثل في توفير هيكل وساطة لمصادر الفوقية غير المتجانسة الممثلة ب

XML, RDF, RuleML

يعمل على دمج البيانات المختلفة و بالتالي يساعد المستخدم في استخراج المعلومات بطريقة شفافة دون اللجوء إلى مصدرها الأصلي

الكلمات الدالة:البيانات الفوقية، الدمج و التكامل، الوسيط، معالجة الطلبات

## Sommaire

<b>Introduction générale</b>	<b>1</b>
Contexte .....	1
Problématique .....	1
Objectifs.....	2
Organisation de travail .....	2
<b>Chapitre I : Etat de l'art sur l'intégration de données</b>	<b>4</b>
I. Introduction.....	4
II. Définition d'intégration.....	4
III. Les problèmes de l'intégration des données.....	4
III.1-Conflits syntaxiques.....	5
III.2-Conflits Structurels.....	6
III.3-Conflits sémantiques.....	8
IV. L'approche médiateur.....	9
IV.1. Les composant d'un système de médiation.....	9
IV.2. Langage de médiation.....	10
IV.3. Les difficultés d'intégrations dans un système de médiation.....	10
IV.4. Architecture de médiation de base.....	12
V. Conclusion.....	14
<b>Chapitre II. Les modèles de sources de métadonnées XML, RDF et RuleML</b>	<b>15</b>
I. Introduction.....	15
II. Généralités sur les métadonnées.....	15
II.1. Qu'est-ce qu'une métadonnée ? .....	15
II.2. Typologie des métadonnées.....	15
II.3. Classification des métadonnées selon la notion de médiateur.....	9
II.3.1. Métadonnées au niveau des sources.....	19
II.3.2. Métadonnées au niveau de la médiation.....	19

II.3.3. Métadonnées entre la médiation et les sources.....	19
III. Les modèles de représentation des métadonnées.....	19
III.1. Le Standard Dublin Core.....	20
III.2. Les modèles d'implémentation.....	22
III.2.1. RDF (Resource Description Framework).....	22
III.2.2. Comment décrire une ressource en utilisant des métadonnées Dublin Core et une syntaxe RDF/XML?.....	27
III.2.3. Le model de données XML.....	29
III.2.4. Les langages de définition d'un document XML.....	31
III.2.4.1. Les DTD.....	31
III.2.4.2. Le schéma XML (XMLS) .....	32
III.2.5. RULEML (Rule Markup Language) .....	33
IV. Conclusion.....	36
<b>Chapitre III: Analyse des besoins et conception</b>	<b>37</b>
Introduction.....	37
Le Cycle de vie d'un logiciel et le langage de modélisation UML.....	37
II.1. Le cycle de vie d'un logiciel.....	37
II.1.1. Définition.....	37
II.1.2. Modèle de cycle de vie en cascade.....	37
II.1.3. Les activités.....	38
1. Spécification des besoins et analyse.....	38
2. Conception.....	39
3. Implémentation.....	40
4. Test.....	40
5. Maintenance.....	40
II .2. Le langage UML.....	40
II .2.1 Définition.....	40



III. Spécification des besoins.....	42
III .1. Recueil des Besoins Fonctionnels.....	42
III.1.1. Identification des Acteurs.....	42
III.1.2. Identification des cas d'utilisation.....	43
III .1.2.1. Diagramme de cas d'utilisation.....	43
1. Diagramme de cas d'utilisation général.....	44
2. Diagramme de cas d'utilisation détaillé pour l'administrateur.....	46
IV. Analyse de système.....	47
IV .1 Diagramme d'activités.....	47
1. Diagrammes d'activités pour l'utilisateur.....	47
2. Diagrammes d'activités pour le mediateur.....	48
3. Diagrammes d'activités pour l'adaptateur.....	50
4. Diagrammes d'activités pour l'administrateur.....	50
IV .2 Diagramme de séquence.....	56
IV .2 .1 Présentation des scenarios et diagrammes de séquence du Système.....	57
1. Authentification.....	57
2. Traiter la requête.....	58
3. ajouter une métadonnée.....	60
4. Modifier les métadonnées.....	62
5. Supprimer une métadonnée.....	63
V. Conception.....	65
V.1. Identification des classes candidates.....	65
V.2. Diagramme de classes.....	65
VI. Architecture générale de notre système de médiation.....	67
VI.1 Description des couches de système.....	67
VI.1.1 Couche Source de Données .....	67
VI.1.2 Couche Adaptateur .....	73

VI.1.3 Couche Médiateur.....	74
VI.1.4 Couche utilisateur.....	76
VI.2 Description des modules.....	76
VII. Conclusion.....	81
<b>Chapitre IV: Implémentation et Test</b>	<b>82</b>
I. Introduction.....	82
II. Réalisation.....	82
II.1. Implémentation des sources.....	82
II.1.1. Classes et hiérarchies des classes.....	82
II.1.2. Source de métadonnées RDF.....	83
II.1.2.1. Altova SemanticWorks 2011.....	83
II.1.2.2. La représentation de la source de métadonnées RDF sur l'éditeur Altova SemanticWorks 2011.....	83
II.1.3. Source de métadonnées XML.....	84
II.1.3.1. Stylus Studio 2011 XML Entreprise.....	84
II.1.3.2. La représentation de la source de métadonnées XML sur l'éditeur Stylus Studio 2011.....	85
II.1.4. Source de métadonnées RuleML.....	85
II.1.4.1. DR-Device.....	86
II.1.4.2. La représentation de la source de métadonnées RuleML sur l'éditeur DR-Device.....	86
II.2. Mécanismes de manipulation des documents XML.....	86
II.2.1. Modèle objet de document: DOM.....	86
II.2.2. Interrogation de documents XML en utilisant XQuery.....	87
II.3. Le langage de programmation JAVA.....	88
II.3.1. Java est un langage orienté objet.....	88
II.3.2. Le SGBD MYSQL.....	89
II.4. Démonstration.....	89

III. CONCLUSION.....	100
----------------------	-----

**Conclusion générale**

## *Liste des tableaux*

Tableau II.1: Éléments de base du Dublin Core et leur signification.....	22
Tableau II.2: Classes RDF/RDFS.....	26
Tableau II.3: Propriétés RDF/RDFS.....	27
Tableau III.1: Les acteurs du système.....	42
Tableau III.2: Description des besoins fonctionnels.....	43
Tableau III.3: Modèle de représentation des descriptions détaillées des cas d'utilisations.....	44
Tableau III.4 : Description du cas d'utilisation « Authentification » .....	45
Tableau III.5 : Description du cas d'utilisation « gérer les métadonnées » .....	45
Tableau III.6 : Description du cas d'utilisation « créer des vues sur les sources » .....	46
Tableau III.7 : Description du cas d'utilisation « créer une vue sur le schéma global ».....	47

## *Liste des figures*

Figure I.1 : Classification des conflits de données.....	5
Figure I.2 : Exemple représente le conflits schématiques.....	7
Figure I.3 : Comparaison des architectures GAV et LAV.....	11
Figure I.4: Architecture générale d'un système de médiation.....	13
Figure II.1 : Typologie des métadonnées.....	16
Figure II.2: Exemple de graphe RDF.....	24
Figure II.3: présentation de graphe RDF dans un fichier XML.....	25
Figure II.4: présentation de règles dans RuleML sous une de ses syntaxes XML.....	34
Figure II.5: présentation de l'exemple précédent par graphe RDF.....	35
Figure II.6: Hiérarchie des règles dans RuleML-Vue graphique.....	35
Figure III.1 : Le cycle de vie de modèle en cascade. ....	38
Figure III.2 : Diagramme de cas d'utilisation globale.....	44
Figure III.3 : Diagramme de cas d'utilisation détaillé pour l'administrateur. ....	46
Figure III.4 : diagramme d'activité du cas d'utilisation « Traiter la requête» ....	48
Figure III.5 : diagramme d'activité du processus « Décomposer la requête» ....	49
Figure III.6 : diagramme d'activité du processus « Recomposer les résultats».....	50
Figure III.7 : diagramme d'activité du cas d'utilisation « authentification ».....	51
Figure III.8 : diagramme d'activité du cas d'utilisation « créer le schéma global».....	52
Figure III.9 : diagramme d'activité du cas d'utilisation « Gérer les métadonnées».....	53
Figure III.10 : diagramme d'activité du cas d'utilisation « Ajouter une métadonnée»..	54

Figure III.11 : diagramme d'activité du cas d'utilisation « modifier les métadonnées».	55
Figure III.12 : diagramme d'activité du cas d'utilisation « supprimer une métadonnée».	56
Figure III.13 : Diagramme de séquence de processus d'authentification (cas normal).	57
Figure III.14 : Diagramme de séquence de processus d'authentification (cas d'erreur).	58
Figure III.15 : Diagramme de séquence de processus « Traiter la requête » (Détailé).	60
Figure III.16 : Diagramme de séquence de processus « Ajouter une métadonnée ».	61
Figure III.17 : Diagramme de séquence de processus « Modifier les métadonnées ».	63
Figure III.18 : Diagramme de séquence de processus « Supprimer une métadonnée ».	64
Figure III.19 : Diagramme de classe de notre système.	66
Figure III.20. Architecture de notre système d'intégration.	67
Figure III.21. Architecture de source RDF.	68
Figure III.22. Architecture de source XML.	69
Figure III .23 : Fragment de la source RuleML.	70
Figure III .24 : Un fragment de Schéma local de la base XML.	71
Figure III .25 : Un fragment Schéma local de la source RuleML.	72
Figure III .26 : Un fragment Schéma local de la source RDF.	73
Figure III .27 : Un fragment Schéma global.	75
Figure VI .28 : Génération du schéma Global.	77
Figure IV.1 : Le contenu de notre Source de métadonnées RDF sur Altova SemanticWorks 2011.	84

Figure IV.2 : Le contenu de notre Source de métadonnées XML sur Stylus Studio 2011.....	85
Figure IV.3 : Le contenu de notre Source de métadonnées RuleML sur DR-Device....	86
Figure IV.4 : page d'accueil de système.....	89
Figure IV.5 : page d'authentification. ....	90
Figure IV.6 : page d'accueil de l'administrateur. ....	91
Figure IV.7 : Créer un schéma global. ....	91
Figure IV.8: génération de schéma global.....	92
Figure IV.9: page pour poser une requête. ....	93
Figure IV.10: page pour poser une requête par domaine.....	93
Figure IV.11: voir la requête générée.....	94
Figure IV.12: voir le résultat de la requête générée. ....	96
Figure IV.13: page pour poser une requête par Examen. ....	96
Figure IV.14: voir la requête générée. ....	97
Figure IV.15: voir le résultat de la requête générée.....	97
Figure IV.16: page pour poser une requête par Test. ....	98
Figure IV.17: voir la requête générée. ....	98
Figure IV.18: voir le résultat de la requête générée. ....	99
Figure IV.19: mise à jour des métadonnées.....	100

# Introduction Générale



## Introduction générale

### 1. Contexte

Le terme « métadonnées » est utilisé de façon générale pour parler d'un ensemble structuré d'information servant à décrire une ressource. Ces informations sont représentées et stockées dans une multitude de sources de données et de façon hétérogène. Le besoin essentiel est donc de pouvoir interroger ces différentes sources de données simultanément et de donner l'impression à l'utilisateur qu'il interroge une unique source de données. De tels systèmes de médiation offrent à l'utilisateur une vue uniforme et centralisée de ces données, cette vue pouvant aussi correspondre à une vision plus abstraite, condensée, qualitative des données et donc, plus significative pour l'utilisateur. Ces systèmes de médiations sont très utiles, en présence de données hétérogènes, car ils donnent l'impression d'utiliser un système homogène.

Les métadonnées permettent de donner du sens au contenu des sources de manière à ce que leur localisation et interrogation soient plus aisées et plus pertinentes. Pour que ces métadonnées soient utilisables, il faut les modéliser dans un format tel que le XML, RDF, et RuleML.

C'est dans le cadre de l'intégration des métadonnées que s'inscrit notre projet « *Intégration de sources de métadonnées hétérogènes XML, RDF, et RuleML par l'approche médiateur* ».

### 2. Problématique

L'accès transparent aux ressources et de manière plus générale constitue un des challenges actuels majeurs de l'informatique.

Intégrer les sources de métadonnées dans le but de fournir aux utilisateurs une interface d'accès uniforme est une tâche difficile ; l'hétérogénéité, la quantité, la dispersion, et la volatilité des ressources constituent les aspects de cette difficulté des systèmes d'intégration.

Les difficultés principales d'intégration qu'on doit la surmonter sont :

*l'approche s'appelle l'approche virtuelle  
concernant à les entités*

- L'hétérogénéité des sources (XML, RDF, et RuleML), leur mise en œuvre pose un certain nombre de problèmes, tant en ce qui concerne la génération des liens sémantique entre le schéma de médiation et les sources de métadonnées (requête de médiation), qu'en ce qui concerne l'adaptation de l'accès aux besoins des utilisateurs.
- La définition du schéma global et la définition de mapping qui reliant le schéma globale aux sources de données.

*répétition*

De manière générale, la problématique peut se résumer dans l'hétérogénéité de métadonnées qu'il est nécessaire de résoudre pour permettre de mettre en correspondance les sources et autoriser l'interrogation et la réponse aux requêtes de façon transparentes.

### 3. Objectifs

Réaliser un système d'intégration qui permet d'accéder à des données à travers une interface uniforme, sans se soucier de leur structure ni de leur localisation.

Notre objectif est de concevoir et de réaliser un système d'intégration de métadonnées hétérogènes, présentées par le modèle XML, RDF et RuleML. Pour permettre à un simple utilisateur de trouver des réponses à ses requêtes en lui cachant l'hétérogénéité de ces sources. Notre système d'intégration sera basé sur l'approche médiateur, en construisant d'une manière automatique un schéma global qui est présenté par le modèle XML.

### 4. Organisation de travail

Pour mieux cerner le problème d'intégration de source de métadonnées hétérogènes, nous avons adopté une méthodologie progressive. Cette méthodologie est envisagée par :

- **Chapitre I: Etat de l'art sur les méthodes d'intégrations de données :** Dans ce chapitre nous avons présenté les différents problèmes d'intégrations ensuite nous avons présentés l'approche médiateur.

➤ **Chapitre II: Etat de l'art sur les modèles de sources de métadonnées:** Dans ce chapitre on a défini les différents types de métadonnées, ainsi que les trois modèles de sources de métadonnées (XML, RDF, RuleML) que nous avons utilisé pour représenter ces dernières.

➤ **Chapitre III: Analyse des besoins et conception :** Ce chapitre comporte la conception de notre système de médiation par le langage de modélisation UML, ainsi qu'une architecture générale de notre système.

➤ **Chapitre V: Implémentation et Test:** Dans ce chapitre nous décrivons les différents outils utilisés dans notre projet, ensuite nous présenterons une démonstration globale de différentes tâches à accomplir par notre application.

➤ **Conclusion générale:** Nous y présenterons les résultats et les perspectives de notre travail.

# ***CHAPITRE I***

## *Etat de L'art sur les Méthodes d'intégration*

## **I. Introduction**

Les informations présentes dans un système d'information sont représentées et stockées dans une multitude de sources de données et de façon hétérogène. Le besoin essentiel est donc de pouvoir interroger différentes sources de données simultanément et de donner l'impression à l'utilisateur qu'il interroge une unique source de données.

## **II. Définition d'intégration**

L'intégration des données est un processus qui vise à extraire et combiner les données provenant de différentes sources en créant un schéma commun de ces sources ou en générant une source de données plus complète en combinant des données de différentes sources afin de répondre à des besoins spécifiques [ANT, 2008].

## **III. Les problèmes de l'intégration des données**

Les données de différents systèmes d'intégration sont, pour la plupart, issues des différentes sources de données hétérogènes, il existe différentes raisons qui expliquent cette hétérogénéité, par exemple différences dans le matériel, le système logiciel, les systèmes des SGBD, et les données... etc.

Le schéma suivant présente les conflits de données en trois niveaux : conflits syntaxiques, conflits structurels et conflits sémantiques [BAL, 2007].

```
<!DOCTYPE Thèse[
<!ELEMENT Thèse (Numthese,titre_t,domaine_t)>
<!ELEMENT Numthese(#PCDATA)>
<!ELEMENT titre_t(#PCDATA)>
<!ELEMENT domaine_t(#PCDATA)>].
```

- Schéma orienté objet: Public Thèse (String Numthese, String titre\_t, String domaine\_t).

### III.2. Conflits Structurels

Résultant d'une structuration et classification différentes des informations.ils sont étroitement liés aux choix de conception.

*Exemple* : représentation de l'information « Téléphone » de manière différente dans le même modèle(Relationnel).

Les numéros de téléphone sont représentés par des attributs dans la relation

Entreprise (Code, Nom, Adresse, tél1, tél2, tél 3, Fax, e\_mail).

Les numéros de téléphone sont regroupés dans une relation à part :

Entreprise (Code, Nom, Adresse, Fax, e\_mail).

Tél\_Entreprise(Code, Tél).

#### *- Les conflits de schémas*

Résultent de l'utilisation de différents concepts pour représenter le même objet.

Une information peut être représentée par une entité dans un système (S1) et par une relation dans autre système (S2).

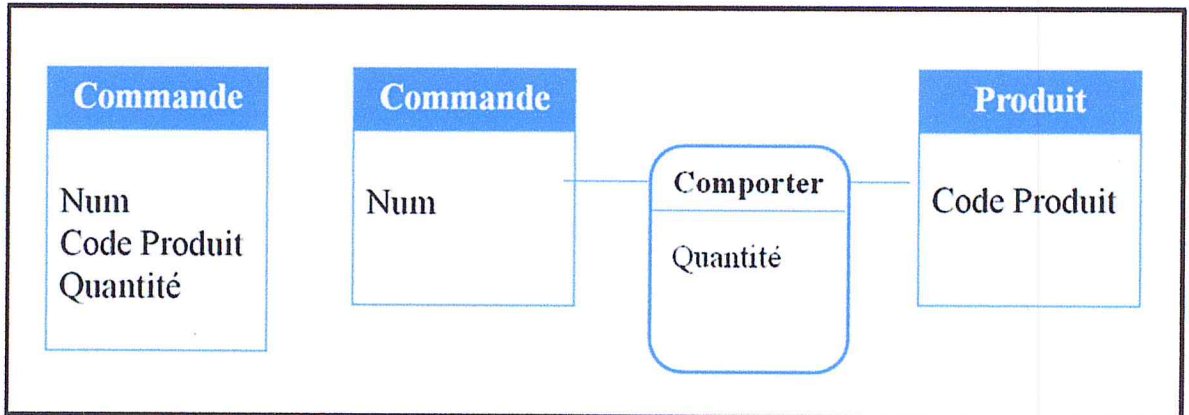


Figure I.2 : Exemple représente le conflits Schématiques.

**- Les conflits de généralisation /spécialisation**

Résultant des différences de hiérarchisation des informations.

*Exemple :* dans un système(S1) les personnes sont regroupées dans un seul objet PERSONNE alors que dans (S2) on utilise deux objets HOMME et FEMME pour représenter les personnes.

**- Les conflits d'agrégation**

Résultant d'un niveau de granularité différent de deux systèmes.

La valeur d'un attribut sur un système correspond à une agrégation des valeurs de plusieurs attributs sur un autre.

*Exemple :* nous disposons des moyennes modulaires des étudiants dans un système(S1) et le détail des notes obtenues au cours des examens dans le deuxième(S2).

**- Les conflits de typage**

Résultant des différences de typage pour le même objet dans les différents systèmes de la coopération.

*Exemple :* le mois est représenté par une chaîne de caractère sur le système (S1), est représenté comme entier sur le système(S2).

***- Les conflits de complétude***

Apparaissent lorsque des objets se correspondent partiellement sur les différents systèmes.

C'est-à-dire qu'une partie d'un objet de système(S1) trouve une correspondance sur le système(S2).

***Exemple***

(S1) : ETUDIANT (matricule, Nom, Prénom, Adresse).

(S2) : STAGIAIRE (N°ins, Nom, Adresse, Ville, Pays).

Dans (S1) l'attribut Adresse contient Rue, Ville et Pays alors que dans(S2) l'adresse contient

Seulement la Rue, Dans (S2) le nom contient au nom et prénom.

**III.3. Conflits sémantiques**

Proviennent des différences d'interprétation des informations partagées entre différents domaines d'application. Plusieurs types de conflits sémantiques apparaissent.

***- Les conflits de noms***

Se trouvent lorsque les différents schémas utilisent des noms différents pour représenter le même concept ou propriété (synonyme), ou des noms identiques pour des concepts ou des propriétés différents (homonymes).

***- Les conflits de contexte***

Se retrouvent dans le cas où les concepts semblent avoir la même signification dans deux schémas mais sont différents à cause de leur contexte. par exemple le poids d'une personne dépend de la date où elle se pèse.



pertinentes et de la résolution des conflits (structurels et sémantiques), chargé de répondre à des besoins à partir des connaissances mises à sa disposition [BAL, 2007].

Pour accéder aux sources de données locales, le médiateur fait appel à des wrappers.

#### *Wrapper*

Un wrapper de données est un logiciel qui :

- ✓ convertit les requêtes exprimées dans le modèle de médiation provenant d'un ou de plusieurs médiateurs vers les modèles de sources de données locales
- ✓ convertit les données résultantes d'une requête, du modèle de base de données locale vers le modèle de médiation
- ✓ un wrapper de données offre une interface homogène locale d'accès aux données sources (résolution des conflits syntaxiques).

#### **IV.2. Langage de médiation**

La médiation adopte un langage unique appelé langage de médiation pour permettre aux différents utilisateurs et applications d'interroger les différentes sources de données de manière uniforme, en masquant les détails d'hétérogénéité et de localisation [BAL, 2007].

#### **IV.3. Les difficultés d'intégrations dans un système de médiation**

On peut classer les difficultés rencontrées dans un système de médiation en deux catégories: l'intégration de schéma et l'évaluation de la requête.

##### *Intégration de schéma*

On peut classer les systèmes d'intégration de données suivant la relation entre les schémas des sources locales par rapport au schéma unifié global sur le médiateur. On distingue deux types de système:

1) *approche descendante (Global As View ou GAV)* où chaque objet du schéma global est défini par une requête sur les sources.

2) *approche ascendante (Local As View ou LAV)* où chaque objet d'une source de données est défini par une requête sur le schéma global.

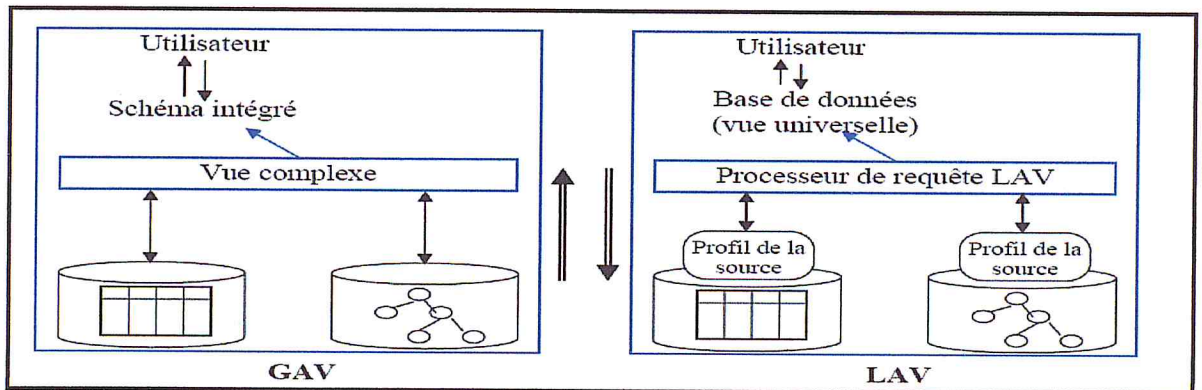


Figure I.3 : Comparaison des architectures GAV et LAV [BAK, 2006].

#### ❖ *Evaluation de requêtes*

Le médiateur présente des vues intégrées des sources de données. Ainsi une requête formulée au médiateur est posée indépendamment des localisations des différentes données intervenant pour calculer le résultat. Cela introduit trois difficultés [BAK, 2006]:

- *la décomposition d'une requête:* il s'agit à partir d'une requête posée sur une vue intégrée, de localiser les données intervenant dans sa résolution, de produire des sous-requêtes spécifiques à chacune des sources, d'ordonner ces sous requêtes et éventuellement d'introduire des opérateurs au niveau du composant de médiation afin de compléter cet ensemble de sous-requêtes. La localisation des sous-requêtes nécessite des structures spécifiques de gestion de métadonnées.

- *la recombinaison des résultats:* une fois les sous-requêtes soumises à chacune des sources, il s'agit de savoir recombinaison les différents résultats entre eux. Les résultats de chacune des sous-requêtes peuvent éventuellement faire l'objet d'un

traitement additionnel soit parce que l'évaluateur de sous-requête n'a pas la capacité nécessaire pour traiter entièrement cette dernière, soit parce que les sous-requêtes comportent des dépendances entre elles.

- *l'optimisation du traitement*: Le médiateur a rarement une vision sur la façon dont sont traitées les sous-requêtes au niveau des sources (placement des données, type de stockage, indexation, stratégie d'évaluation). De plus la distribution des données sur des sources disjointes ne permet pas d'utiliser directement les algorithmes employés normalement dans le cas d'un SGBD centralisé.

#### **IV.4. Architecture de médiation de base**

Le concept de médiateur dans le domaine des systèmes d'information est apparu pour intégrer des sources d'informations hétérogènes. "Gio Wiederhold" a proposé une architecture logicielle à trois niveaux et a défini un médiateur de la façon suivante: "un médiateur est un module logiciel qui exploite la connaissance de certains ensembles ou sous-ensembles de données pour créer de l'information pour des applications à un niveau supérieur"[BAK, 2006]. Cette architecture, qui c'est largement imposée dans les systèmes d'information, repose sur les trois couches suivantes: sources, médiateurs et clients.

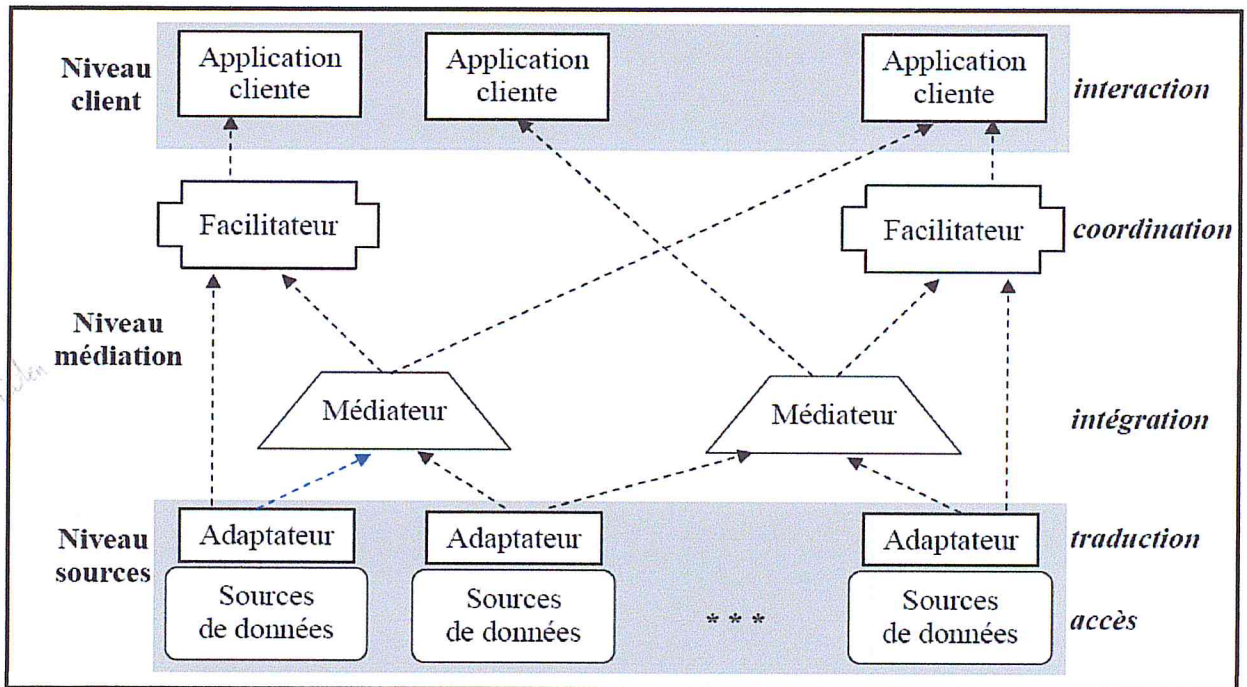


Figure I.4: Architecture générale d'un système de médiation [BAK, 2006].

▪ **Le niveau sources**

Comporte les différentes sources de données ; à l'aide d'un adaptateur (wrapper), il est capable de communiquer avec le médiateur et facilitateur du niveau supérieur, en leur fournissant une vue homogène de la source à laquelle il est associé. Un adaptateur accepte une requête donnée dans le langage commun du médiateur, la transcrit dans le langage natif de la source et exécute la requête. Le résultat de la requête transmis sous forme native est alors transformé suivant le modèle de données global du médiateur et renvoyé à celui-ci.

▪ **Le niveau médiateur**

Comporte des médiateurs permettant d'intégrer les données en provenance de différentes sources afin de répondre aux requêtes des utilisateurs. Ce module joue un rôle actif dans la couche entre les applications utilisateurs et les sources de données. Son rôle est de fournir à la couche supérieure une vue centralisée des sources qui sont hétérogènes et distribuées.

On trouve aussi à ce niveau des facilitateurs permettant d'identifier les sources qui peuvent être des sources de données ou des médiateurs, et de composer les réponses pour les utilisateurs.

- **Le niveau clients** : comporte les applications clientes (navigateurs, programmes d'application, interface graphique).

## **V. Conclusion**

Les systèmes d'intégration sont basés sur le mécanisme de vues pour présenter une vue unifiée de données provenant de sources hétérogènes. Cette vue unifiée est appelée schéma médiateur du système.

Il existe principalement deux approches pour construire le schéma médiateur d'un système d'intégration. L'approche GAV et l'approche LAV.

Lorsque les vues sont virtuelles, on parle de système médiateur, ces vues sont extraites de différentes sources de données (base de données, fichier texte, page web ...etc.), dans notre cas on a 3 modèles de sources métadonnées : XML, RDF, et RuleML qu'on va les étudier dans le chapitre suivant.

## **I. Introduction**

La maîtrise sur des grands ensembles d'information devient de plus en plus complexe et de plus en plus fastidieux. Les métadonnées constituent une voie pour aider l'utilisateur ou le gestionnaire d'information à comprendre, retrouver, comparer des informations sans forcément avoir recours directement au contenu de celles-ci. En effet, les métadonnées peuvent être vues comme étant des données structurées qui décrivent les données et qui peuvent s'appliquer à tous types de données.

L'objectif de ce chapitre est de définir les différents modèles de source de métadonnées (XML, RDF, RuleML) quand va-les utilisé dans notre projet.

## **II. Généralités sur les métadonnées**

### **II.1. Qu'est-ce qu'une métadonnée ?**

Une métadonnée est une donnée à propos d'une autre donnée. En sciences de l'information, les métadonnées sont « *des ensembles de données structurées décrivant des ressources physiques ou numériques, ou, sur un plan plus fonctionnel, de l'information structurée qui décrit, explique, localise la ressource et en facilite la recherche, l'usage et la gestion* » [MBA, 2008].

Les métadonnées peuvent être incluses dans la ressource elle-même ou enregistrées dans un fichier séparé selon le type du contenu, comme c'est le cas pour une notice dans un catalogue de bibliothèque [DIA ,2001].

### **II.2. Typologie des métadonnées**

Le terme « **métadonnée** » regroupe plusieurs typologies. Cette diversité est relative à plusieurs critères.

*Anne J. Gilliland-Swetland* donne un résumé intéressant sur ces différentes typologies, dans son article : Introduction to Metadata.

La figure suivante représenté ces différentes typologies :

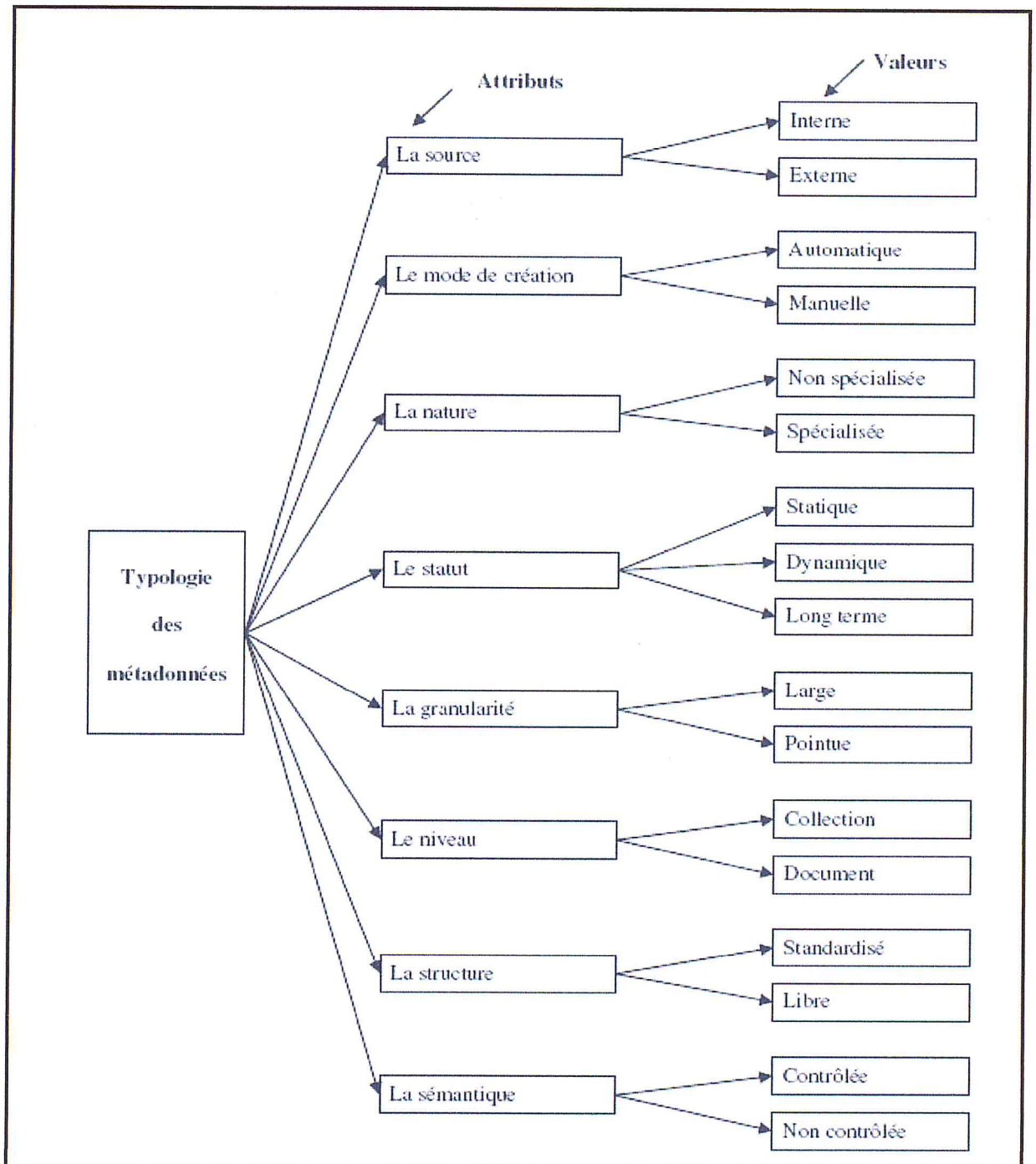


Figure II.1 : Typologie des métadonnées [GIL, 2000].

### *La source*

Cet attribut peut prendre deux valeurs :

➤ **Métadonnée interne**, générée au moment de la création du document, elles sont intégrées dans la ressource elle-même, de façon implicite ou explicite.

- **Implicite** : Le logiciel génère automatiquement des informations sur le document. *à TD explicite*
- **Explicite** : Sous forme de balisage de données : on inclut un ou plusieurs jeux de métadonnées dans la ressource. *même*

➤ **Métadonnée externe** :

Elles sont contenues soit dans une notice séparée du document, comme c'est le cas d'une notice d'un catalogue de bibliothèque ; soit dans un thesaurus ou une base de données externe via des outils pour constituer un réservoir de métadonnées. Le but est d'assister l'utilisateur dans sa recherche d'information et sa navigation. (

### *Le mode de création*

Nous distinguons principalement deux modes de création :

- **Automatique**: Les métadonnées sont générées automatiquement par le système informatique « indexation et condensation automatiques ».
- **Manuel** : Les métadonnées sont générées à partir de schémas de métadonnées standardisés, tel que le Dublin Core.

### *La nature*

- **Non spécialisée**: Les métadonnées sont créées par des non spécialistes du domaine.
- **Spécialisée**: Les métadonnées sont générées par des bibliothécaires, archivistes ou professionnels de l'information.



### *Le statut*

- **Statique:** Les métadonnées accompagneront le document tout au long de son cycle de vie. Exemple : titre, provenance, date de création.
- **Dynamique:** Les métadonnées peuvent évoluer en fonction du cycle de vie : structure des répertoires, résolution des images, information sur les modes d'accès.
- **Long Terme:** Les métadonnées accompagneront le processus de préservation : informations procédurales et techniques, informations légales, documentation sur la gestion à long terme.

*La granularité peut se faire en 25 mot-large pour hiérarchique*

- **Large:** La description du document sera exhaustive (25 mots clé, par exemple), ce qui aura une incidence au niveau du repérage de l'information, augmentation du bruit et diminution de la pertinence. Il y aura également incidence sur les modes de production (coûts plus élevés) et les modes de gestion (stockage, élimination...). Le choix de la granularité doit se faire en fonction des besoins informationnels.
- **Pointue:** La description du document sera pointue (5 à 10 mots clé, par exemple). *5 ou 6 mot pour decrive*

### *Le niveau*

- **Collection:** Les métadonnées sont attribuées pour définir et préserver une collection. *pour preser une collection de doc*
- **Document:** Les métadonnées sont attribuées pour définir et préserver un document. *pour preser un document complet*

### *La structure*

- **Standardisée:** tel que : La DTD (HTML, TEI).  
Les Métalangages (XML, SGML).  
Les ensembles de métadonnées structurées (Dublin Core).
- **Libre:** tel que les systèmes offerts par les logiciels.  
*ne doit avoir ces propriétés par exemple*

### *La sémantique*

- **Contrôlée:** La syntaxe est définie par le standard, tel que Dublin Core, MARC ou USMARC.
- **Non contrôlée:** Comme les descripteurs dans les balises META en HTML.

## **II.3. Classification des métadonnées selon la notion de médiateur**

### **II.3.1. Métadonnées au niveau des sources**

Les métadonnées définies au niveau des sources décrivent le schéma de chaque source de données, l'ensemble des relations sources appartenant à chaque schéma source, les clés des relations, les attributs de chaque relation, les assertions entre les relations.

### **II.3.2. Métadonnées au niveau de la médiation**

Les métadonnées définies au niveau de la médiation caractérisent le schéma de médiation, l'ensemble des relations de médiation appartenant à ce schéma de médiation, les clés des relations, les dépendances fonctionnelles éventuelles, et les attributs de chaque relation.

L'ensemble d'assertions définies sur une relation de médiation est composé essentiellement de dépendances fonctionnelles qui relient l'attribut clé aux attributs non clés.

### **II.3.3. Métadonnées entre la médiation et les sources**

Les métadonnées entre la médiation et les sources sont des correspondances linguistiques reliant un attribut d'une relation de médiation à un attribut d'une relation source.

## **III. Les modèles de représentation des métadonnées**

Plusieurs standards de métadonnées sont reconnus et utilisés aujourd'hui, Chacun répond à des besoins spécifiques, et ils sont en majorité nécessaires et complémentaires parmi ces standards il y'a le standard *Dublin Core*.

### III.1. Le Standard Dublin Core

#### *Définition*

DC (Dublin Core) ou DCMI (Dublin Core Metadata Initiative) a été proposé pour faciliter la recherche de ressources peu complexes. Il est défini comme un vocabulaire standard et sémantique, qui permettant la description des métadonnées.

#### *Caractéristique du DC*

Le Dublin Core tente de concilier les caractéristiques suivantes [DIA ,2001]:

✚ **La simplicité de création et de gestion:** L'ensemble des éléments du Dublin Core a été tenu aussi sommaire et simple que possible afin de permettre au non-spécialiste de créer des notices descriptives simples pour les ressources informationnelles, de façon facile et économique et ce, tout en permettant des recherches efficaces de ces ressources.

✚ **Une sémantique communément comprise:** La découverte d'information est gênée par des différences de terminologies et de pratiques descriptives d'un champ des connaissances à l'autre. Le Dublin Core peut aider les non spécialiste à trouver son chemin en supportant un ensemble commun d'éléments dont la sémantique est universellement comprise.

✚ **Envergure internationale:** L'ensemble d'éléments du Dublin Core a été d'abord développé en anglais mais des versions sont créées en plusieurs autres langues. En novembre 1999, il y avait des versions en plus de 20 langues incluant le finnois, le norvégien, le japonais, le français, le portugais, l'allemand, le grec, l'indonésien et l'espagnol...etc. Le groupe de travail sur le Dublin Core multilingue coordonne les efforts pour lier ces versions dans un registre distribué utilisant la technologie du *Resource Description Framework* actuellement en développement au Consortium *World Wide Web (W3C)*.

✚ **L'extensibilité:** Les développeurs du Dublin Core ont reconnu l'importance de prévoir un mécanisme permettant d'étendre l'ensemble des éléments du DC pour d'autres besoins de découvertes de ressources. On s'attend à ce que d'autres communautés d'experts en métadonnées créent et administrent d'autres ensembles de métadonnées. Le présent modèle permet à différentes communautés d'utiliser l'ensemble des éléments du DC pour la description primaire de l'information, qui devient alors utilisable, tout en permettant des ajouts, spécifiques à un domaine, qui soient pertinents dans une communauté particulière.

### *Liste des éléments du Dublin Core*

Le **Dublin Core** comprend 15 éléments de description qui sont classés en trois groupes [GAY, 2006]:

- ceux liés au contenu de la ressource: titre, description, sujet, source, couverture, type, relation.
- ceux liés à la propriété intellectuelle: créateur, contributeur, Editeur, droit.
- ceux liés à une instance particulière de la ressource: date, format, identifiant, langue.

### *Description des éléments*

Élément	Description
<b>Title</b>	Il s'agit a priori du titre principal du document.
<b>Creator</b>	Nom de la personne, de l'organisation ou du service à l'origine de la rédaction du document.
<b>Subject</b>	Une phrase, des mots-clés, un code de classification ou toute autre donnée textuelle aidant à décrire le sujet du contenu de la ressource.
<b>Description</b>	Description du document : résumé, table des matières, ou texte libre.
<b>Publisher</b>	Nom de la personne, de l'organisation ou du service à l'origine de la publication du document.

<b>Contributor</b>	Nom d'une personne, d'une organisation ou d'un service qui contribue ou a contribué à l'élaboration du document.
<b>Date</b>	Une date qui est généralement la date de création ou de publication de la ressource. Il est recommandé que cette date suive le format YYYY-MM-DD
<b>Type</b>	Nature ou genre du contenu : La nature ou le genre du contenu de la ressource. En général, il est recommandé ici d'utiliser les mots d'un vocabulaire de classification reconnu.
<b>Format</b>	Le format du support physique associé à la ressource.
<b>Identifiant</b>	Identificateur non ambigu : il est recommandé d'utiliser un système de référencement précis, par exemple les URI ou les numéros ISBN.
<b>Source</b>	Ressource dont dérive le document : le document peut découler en totalité ou en partie de la ressource en question. Il est recommandé d'utiliser une dénomination formelle des ressources, par exemple leur URI.
<b>Language</b>	Langue du contenu intellectuel de la ressource.
<b>Relation</b>	Lien avec d'autres ressources. De nombreux raffinements permettent d'établir des liens précis, par exemple de version, de chapitres, de standard, etc.
<b>Coverage</b>	Couverture spatiale (point géographique, pays, régions, noms de lieux) ou temporelle.
<b>Rights</b>	Les droits concernent la propriété intellectuelle, le copyright, les conditions d'exploitation, de réutilisation.

Tableau II.1 : Éléments de base du Dublin Core et leur signification [CHI, 2010].

## **III.2. Les modèles d'implémentation**

### **III.2.1. RDF (Resource Description Framework)**

#### **Définition**

- ✚ RDF (Resource Description Framework) est un moyen d'encoder, d'échanger et de réutiliser des métadonnées structurées [ELB, 2009].
- ✚ C'est un cadre de description des ressources applicable à n'importe quel domaine [ELB, 2009].
- ✚ Un document RDF est un ensemble de triplets de la forme *< sujet (ressource), prédicat (propriété), objet (valeur) >*. Cet ensemble de triplets peut être représenté de façon naturelle par un graphe RDF [HAC] :
  - ✓ *Un sujet* : n'importe quelle information dont on veut parler.
  - ✓ *Une propriété* : c'est une caractéristique du sujet.
  - ✓ *Un objet* : c'est la valeur de la caractéristique.

### Graphe RDF

Un graphe RDF, c'est une représentation graphique des déclarations RDF. Cette représentation utilise la notion de nœud (pour les ressources et les valeurs) et d'arc (pour les propriétés).

- **Les nœuds** [MBA, 2008]:
  - Chaque *sujet* est un **URI** (Uniform Resource Identifier) ou un **nœud anonyme**.
  - Chaque *prédicat* est un **URI**.
  - Chaque *objet* est un **URI**, un **littéral** ou un **nœud anonyme**.
- **Les arcs** : Un document RDF ainsi formé correspond à un multi graphe orienté étiqueté. Chaque triplet correspond alors à un arc orienté dont le label est le *prédicat*, le nœud source est le *sujet* et le nœud cible est l'*objet* [MBA, 2008].

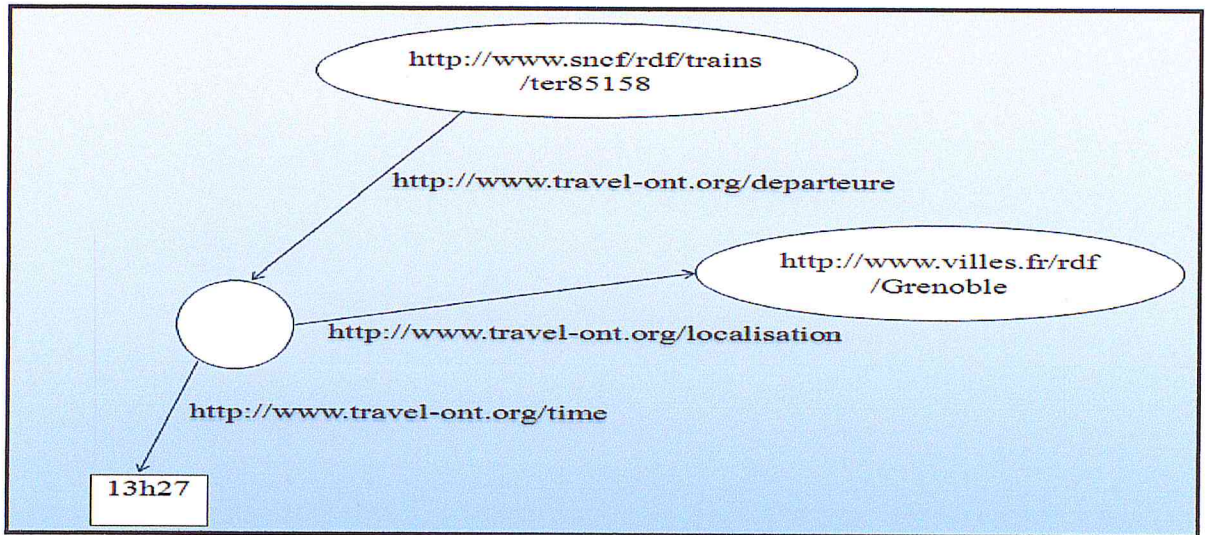


Figure II.2: Exemple de graphe RDF [HAC].

### Description de la Figure II.2

Cette figure présente une partie d'un document RDF (il s'agit d'un exemple fictif, montrant comment la SNCF (Société Nationale des Chemins de Fer) pourrait donner une interface RDF à sa base de données de voyages). Les termes de la forme `http://...` sont des URIs qui identifient des ressources définies de façon unique. Notons dans les URIs que certaines ressources sont spécifiques à la SNCF (le train), et que d'autres (départeur ...) sont issus d'une ontologie dédiée aux voyages. Les objets d'un triplet qui sont des littéraux sont représentés dans un rectangle (ici, `13h27`). Le sommet non étiqueté représente une variable. Intuitivement, ce graphe peut se comprendre comme « le train TER 85158 part de Grenoble à 13h27 » [HAC].

### La syntaxe de RDF

RDF étant un modèle de données sous forme de graphe, peut être représenté de diverses manières (*XML*, *N3*, *N-triples*, *Turtle...etc.*) la plus connue est la syntaxe **RDF/XML** qui, utilisant la métalangue XML, permet de créer des descriptions facilement manipulables par la machine.

- **Syntaxe RDF /XML :**

```
conteneur RDF
<?xml version="1.0"?>
<rdf:RDF utilisation des espaces de noms rdf et dc
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description about précise la ressource à décrire
rdf:about="http://xml.coverpages.org/RadioTV-NewsML-en-
20020224.pdf"> valeur M. Onishi
propriété creator <dc:creator>M. Onishi</dc:creator>
                  <dc:title>RadioTV-NewsML in Japan</dc:title>
                  <dc:date>2002-02-21</dc:date>
                  <dc:type>Text</dc:type>
                  <dc:format>application/pdf</dc:format>
                </rdf:Description>
  </rdf:RDF>
```

Figure II.3: présentation de graphe RDF dans un fichier XML [CHA, 2011].

### Le schéma RDF (RDFS)

RDFS est une prolongation de RDF. Un schéma RDF permet de décrire un vocabulaire et une sémantique des types de propriétés utilisées par une communauté d'utilisateurs [ELB, 2009].

RDFS offre les moyens de définir un modèle (ou bien encore un schéma) de métadonnées qui permet de [PHI, 2004]:

- donner du sens aux propriétés associées à une ressource ;
- formuler des contraintes sur les valeurs associées à une propriété afin de lui assurer aussi une signification.

### Le vocabulaire utilisé dans RDF

Ci-dessous sont présentés deux tableaux qui donnent une vue générale du vocabulaire de RDF, reliant le vocabulaire originellement défini dans la spécification de la syntaxe et du modèle RDF avec les classes et les propriétés qui ont leur origine dans le schéma RDF.



Nom de la classe	Commentaire
<b>rdfs:Resource</b>	La classe Resource, tout.
<b>rdfs:Literal</b>	La classe des valeurs littérales, c'est à dire des chaînes caractères et les entiers.
<b>rdf:XMLLiteral</b>	La classe des valeurs littérales permises par XML.
<b>rdfs:Class</b>	La classe des classes.
<b>rdf:Property</b>	La classe des propriétés RDF
<b>rdfs:Datatype</b>	La classe des types de données RDF.
<b>rdf:Statement</b>	La classe des déclarations RDF.
<b>rdf:Bag</b>	La classe des containers non ordonnés.
<b>rdf:Seq</b>	La classe des containers ordonnés.
<b>rdf:Alt</b>	La classe des containers d'alternatives.
<b>rdfs:Container</b>	La super classe des containers.
<b>rdfs:ContainerMembershipProperty</b>	La classe des propriétés des membres des containers, <code>rdf:_1</code> , <code>rdf:_2</code> , ..., dont tous sont des sous-propriétés de 'member'.
<b>rdf:List</b>	La classe des listes RDF.

Tableau II.2: Classes RDF/RDFS [PHI, 2004].

Nom propriétés	Commentaire	domaine	range
<b>rdf:type</b>	Le sujet (ressource), une instance d'une classe	<code>rdfs:Resource</code>	<code>rdfs:Class</code>
<b>rdfs:subClassOf</b>	Le sujet est une sous-classe d'une classe.	<code>rdfs:Class</code>	<code>rdfs:Class</code>
<b>rdfs:subPropertyOf</b>	Le sujet, une sous-propriété d'une propriété.	<code>rdf:Property</code>	<code>rdf:Property</code>
<b>rdfs:domain</b>	Le domaine des ressources s'appliquant à une ressource. (A domain of the subject property.)	<code>rdf:Property</code>	<code>rdfs:Class</code>
<b>rdfs:range</b>	Le domaine des ressources s'appliquant à une propriété. (A range of the subject property.)	<code>rdf:Property</code>	<code>rdfs:Class</code>

<b>rdfs:label</b>	Nom lisible par un humain du sujet (ressource).	rdfs:Resource	rdfs:Literal
<b>rdfs:comment</b>	Une description du sujet (ressource).	rdfs:Resource	rdfs:Literal
<b>rdfs:member</b>	Un membre de la ressource. (A member of the subject resource.)	rdfs:Resource	rdfs:Resource
<b>rdf:first</b>	Le premier article d'une liste RDF de sujet. (The first item in the subject RDF list.)	rdf:List	rdfs:Resource
<b>rdf:rest</b>	Le reste des articles d'une liste RDF de sujet après le premier article. (The rest of the subject RDF list after the first item.)	rdf:List	rdf:List
<b>rdfs:seeAlso</b>	Information complémentaire sur le sujet.	rdfs:Resource	rdfs:Resource
<b>rdfs:isDefinedBy</b>	La définition du sujet.	rdfs:Resource	rdfs:Resource
<b>rdf:value</b>	Propriété idiomatique utilisée pour déclarer des valeurs structurées (i.e. typées).	rdfs:Resource	rdfs:Resource
<b>rdf:subject</b>	Le sujet d'une déclaration (statement) RDF	rdf:Statement	rdfs:Resource
<b>rdf:predicate</b>	Le prédicat d'une déclaration (statement) RDF	rdf:Statement	rdf:Property
<b>rdf:object</b>	L'objet d'une déclaration (statement) RDF	rdf:Statement	rdfs:Resource

Tableau II.3: Propriétés RDF/RDFS [PHI, 2004].

### III.2.2. Comment décrire une ressource en utilisant des métadonnées Dublin Core et une syntaxe RDF/XML [FOR, 2009] ?

1. Déclarer que le document est un document RDF. Utiliser une balise entrante pour déclarer l'élément RDF précédé du préfixe de l'espace de noms du RDF (rdf) suivi de deux points « : ». Fermer l'élément par la balise </rdf : RDF >.

```
< rdf : RDF >  
< /rdf : RDF >
```

2. Déclarer l'espace de noms du RDF et le préfixe utilisé pour le désigner. Utiliser l'attribut `xmlns` suivi de deux points « : » du préfixe RDF est de l'URL de l'espace de noms comme de l'attribut.

```
< rdf: RDF xmlns :rdf= " http://www.w3.org/1999/02/22-rdf-syntax-ns# " >  
< /rdf: RDF >
```

3. Déclarer l'espace de noms du Dublin Core dans la balise de l'élément racine RDF. Utiliser l'attribut `xmlns` suivi de deux points « : » du préfixe `dc` est de l'URI de l'espace de noms comme valeur de l'attribut.

```
< rdf :RDF xmlns :rdf= " http://www.w3.org/1999/02/22-rdf-syntax-ns# "  
xmlns :dc="http://purl.org/dc/elements/1.1/" >  
< /rdf :RDF >
```

4. Déclarer l'élément RDF description pour indiquer qu'il sera question d'une description. Utiliser une balise entrante et une balise fermante.

```
< rdf :RDF xmlns :rdf= " http://www.w3.org/1999/02/22-rdf-syntax-ns# "  
xmlns :dc="http://purl.org/dc/elements/1.1/" >  
<rdf:Description >  
</rdf:Description >  
< /rdf :RDF >
```

5. Utiliser l'attribut RDF `about` pour identifier le sujet de la description. Indiquer l'URL de la ressource à décrire comme valeur de l'attribut `rdf : about`.

```
< rdf :RDF xmlns :rdf= " http://www.w3.org/1999/02/22-rdf-syntax-ns# "
xmlns:dc="http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="http://cours.ebsi.umontreal.ca/INU1020/">
</rdf:Description>
< /rdf :RDF>
```

6. Pour la déclaration des propriétés dont la valeur est littérale, utiliser des balises entrantes et fermantes entre lesquelles vous placez la valeur littérale de la propriété. Utilisez une propriété dont l'URI est raccourci (préfixe : nom\_de\_la\_propriété). Définissez la langue de la valeur en utilisant l'attribut XML `xml :lang`.

```
< rdf :RDF xmlns :rdf= " http://www.w3.org/1999/02/22-rdf-syntax-ns# "
xmlns:dc="http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="http://cours.ebsi.umontreal.ca/INU1020/">
<dc:title xml:lang="fr"> Organisation de l'information numérique</dc:title>
</rdf:Description>
< /rdf :RDF>
```

7. Pour la déclaration des propriétés permettant d'établir une relation avec une autre ressource désignée par un URI, indiquer l'URI comme valeur de l'attribut RDF `resource` :

```
< rdf :RDF xmlns :rdf= " http://www.w3.org/1999/02/22-rdf-syntax-ns#
xmlns:dc="http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="http://cours.ebsi.umontreal.ca/INU1020/">
<dc:relation rdf :resource= "http://www.etudes.umontreal.ca/index_fiche_prog/105350_desc.html/">
</rdf:Description>
< /rdf :RDF>
```

### III.2.3. Le model de données XML

#### Définition

XML (EXtensible Markup Language) est un langage de description et d'échange de données semi-structurées, il permet de décrire la structure logique des documents. A l'aide d'un système de balisage, XML permet de marquer les éléments qui composent la structure et les relations entre ces éléments [MBA, 2008].

### **La syntaxe d'un document XML**

XML utilise des balises et des attributs. XML permet à l'utilisateur de définir son propre jeu de balises dans le but de personnaliser la structure des documents [MBA, 2008].

Le standard XML a été conçu pour être utilisée de deux manières distinctes [MBA, 2008]:

- d'une part, nous pouvons utiliser une DTD, qui spécifie la structure logique d'une classe de documents et définit les balises à utiliser pour identifier les entités de cette structure. Le document XML faisant appel à cette DTD est dit "*document valide*" ;
- d'autre part, un document XML peut être écrit sans DTD.

### ***L'entête d'un fichier XML***

```
<? xml version = " 1.0" encoding="ISO-8859-1" standalone=yes ?>
```

Cette expression indique au processus qui va traiter le document : la version du langage XML utilisé dans le document, le codage des caractères utilisés dans le document, si le document XML fait référence à une DTD ou à un schéma xml définit dans autre fichier, il faut mettre "no", sinon on utilise "yes".

### ***Syntaxe des balises*** [CRA, 2002]

- balise ouvrante et fermante : (par exemple : <prenom> Johnny </prenom>).
- balise vide : <balise\_vide/>
- les balises sont imbriquées : (par exemple : <star><prenom> </prenom> <nom> </nom></star>) et doivent être toujours fermées.

- entre les deux balises on trouve le ‘contenu’ de l’élément.
- un élément non vide ‘contient’ éventuellement des éléments ‘enfants’ et/ou éventuellement du texte. Quand il contient les deux, on parle de contenu mixte
- les éléments sont ‘case sensitive’ : (par exemple : <prenom> Johnny </Prenom> : pas bon).
- le premier élément est la ‘racine’ du document.

#### **Document XML bien formé** [CRA, 2002]

Un document XML ‘*bien formé*’ doit respecter les règles suivantes :

- N’avoir qu’une seule racine
- Tous les éléments doivent avoir une balise d’ouverture et une balise de fermeture.
- Les valeurs d’attributs doivent être entre guillemets.
- Les imbrications doivent être rigoureuses (première balise ouverte, dernière fermée).
- Formulation abrégée : la balise vide : <balise\_vide>.
- Pas de blanc à l’intérieur des noms de balise.

#### **III.2.4. Les langages de définition d’un document XML**

Généralement, il existe deux langages pour définir un modèle XML

- *DTD, Document Type Définition* (simplifié pour XML).
- *XML schema* (mieux typé et plus puissant).

##### **III.2.4.1. Les DTD**

###### *Définition*

Le W3C propose une définition de document type appelée **DTD** (Document Type Définition), c’est-à-dire une grammaire permettant de vérifier la validité du document XML.

Une DTD est un fichier permettant de vérifier qu'un document XML est conforme à une structure donnée.

La norme XML n'impose pas l'utilisation d'une DTD pour un document XML, mais elle impose par contre le respect exact des règles de base de la norme XML.

Une DTD peut être définie de 2 façons :

- sous **forme interne**, en incluant la grammaire dans le même document XML ; c'est à dire déclarée le DTD au début du document XML par `< !DOCTYPE nom_racine [description de la DTD] >`.
- sous **forme externe**, soit en appelant un fichier contenant la grammaire à partir d'un fichier local ou bien en y accédant par son URL (adresse web).

#### III.2.4.2. Le schéma XML (XMLS) [CHA, 2004]

Le W3C a proposé un nouveau langage de définition de schéma de documents qu'est XML Schéma pour traiter les déficiences des DTD.

XML Schéma propose, en plus des fonctionnalités fournies par les DTD, plusieurs nouveautés à savoir :

- Un grand nombre de types de données intégrées comme les booléens, les entiers, les intervalles de temps, etc. De plus, il est possible de créer de nouveaux types par ajout de contraintes sur un type existant.
- Des types de données utilisateurs qui nous permettent de créer notre propre type de données nommé.
- La notion d'héritage : les éléments peuvent hériter du contenu et des attributs d'un autre élément. C'est sans aucun doute l'innovation la plus intéressante de XML Schéma, car elle permet la réutilisation.
- Les indicateurs d'occurrences des éléments peuvent être tout nombre non négatif.

- Une grande facilité de conception modulaire des schémas.

### *Les types de données dans XMLS*

#### Types simples

Les types de données simples les plus courants sont représentés par : chaînes de caractères, entiers, dates, réels, etc.

#### Type complexe

Un type de données complexe peut être caractérisé par son modèle de contenu ; c'est-à-dire comment ses sous-éléments sont susceptibles d'apparaître.

Un type complexe est défini à l'aide de l'élément `<xsd:complexType name=" ... ">` qui pourra contenir, entre autres, une autre, une séquence d'éléments, une série d'attributs, etc.

### **III.2.5. RULEML (Rule Markup Language)**

#### **Definition [DIO, 2007]**

- RuleML a vu le jour en Août 2000 lors de la conférence internationale sur l'intelligence artificielle. RuleML a pour objectif de développer un formalisme standard de règles métier neutre et ouvert basé sur XML/RDF dans le but de permettre à des systèmes hétérogènes de s'échanger des règles.
- RuleML est une proposition pour fournir un langage de représentation de règles, à l'origine avec une syntaxe XML.
- La syntaxe de RuleML manipulant des graphes (des ensembles de relations unaires et binaires), La spécification de RuleML propose ainsi plusieurs syntaxes, certaines utilisant uniquement XML ou RDF et certaines hybrides mélangeant XML et RDF.
- RuleML a pour objectif de développer un formalisme standard de règles métier neutre et ouvert basé sur XML/RDF dans le but de permettre à des systèmes hétérogènes de s'échanger des règles.



Les types de règles [DIO, 2007]

RuleML permet de modéliser différents types de règles selon le contexte applicatif :

- des règles réactives, du type " si événement E, alors E' ",
- des règles de transformation, du type " A se transforme en B ",
- des règles d'implication, du type " A implique B ",
- des faits,
- des requêtes,
- des contraintes d'intégrité, du type " A et B et C vrais ".
- des balises XML spécifiques sont introduites pour chaque type de règle.

```
<Atom>
<Rel> spending</Rel>
<Ind>Peter Miller</Ind>
<Ind>min 5000 euro</Ind>
<Ind>previous year</Ind>
</Atom>
```

Figure II.4: présentation de règles dans RuleML sous une de ses syntaxes XML [HAR and all, 2005].

**Description de la Figure :** Cette figure exprimer que "Peter Miller's spending has been min 5000 euro in the previous year." (Peter Miller's dépensé 5000 euro dans l'année passé).

On peut représenter cet exemple par un graphe se forme RDF :

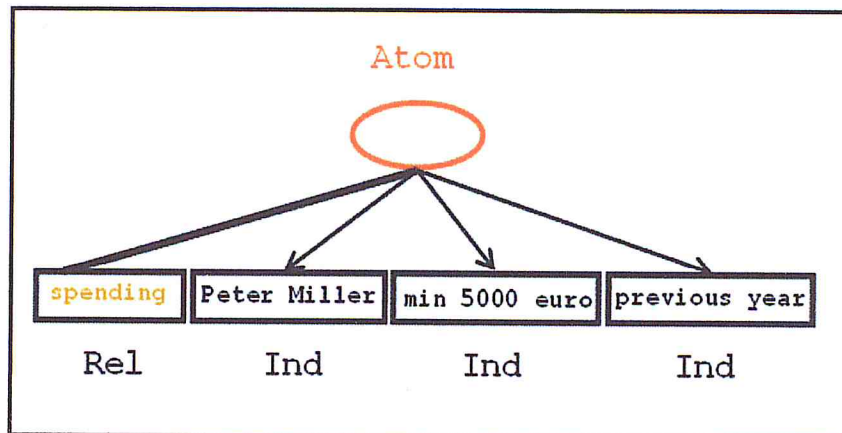


Figure II.5: présentation de l'exemple précédent par graphe RDF [HAR and all, 2005].

L'ovale (Atom) représenté une ressource anonyme et les rectangles représentent les ressources littérales.

### Hierarchie des règles dans RuleML

RuleML couvre la hiérarchie de règles suivante : règles de réaction, règles de transformation, règles de dérivation, règles de fait, règles de requête et règles de contrainte d'intégrité.

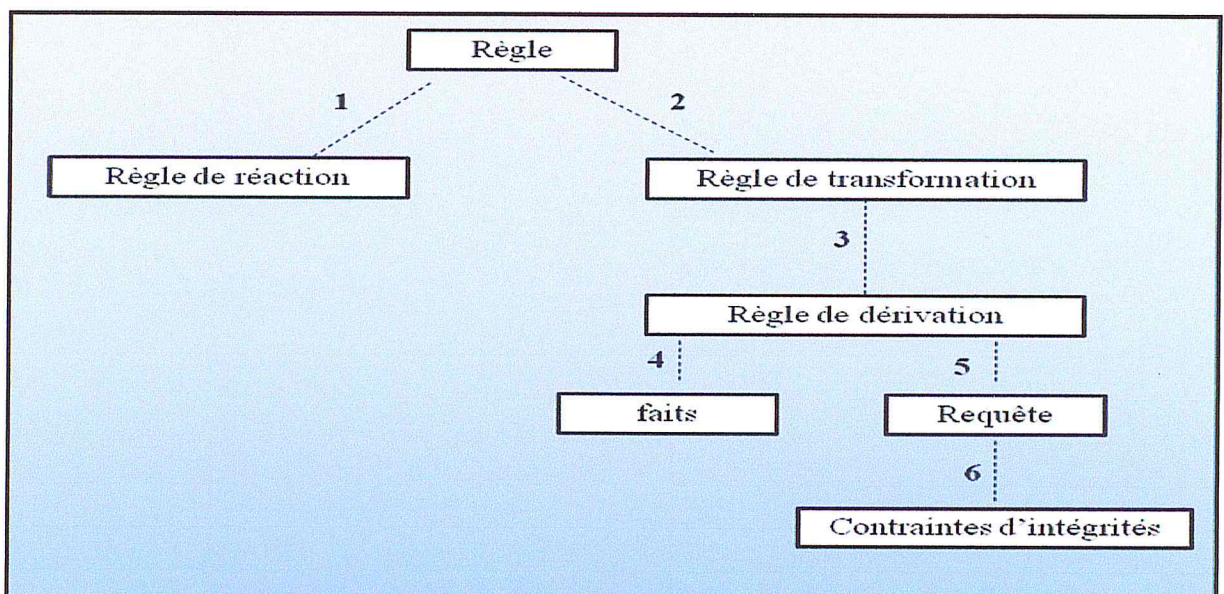


Figure II.6: Hiérarchie des règles dans RuleML-Vue graphique [DIO, 2007].

#### **IV. Conclusion**

Les métadonnées sont des données structurées, standardisées qui décrivent le contenu de document que souhaite partager à des utilisateurs.

Ce chapitre est consacré à la présentation des trois modèles utilisés pour représenter les métadonnées tels que :

- Le *XML*, langage de balises analysable, destiné à une diffusion à grande échelle sur le Web.
- Le *RDF*, langage de représentation de métadonnées sur le web.
- Et en fin le *RuleML*, langage de représentation de règles.

Notre système de médiation permettant d'intégrer ces modèles de métadonnées, afin de construire un schéma global (médiateur), qu'on va les étudier dans le chapitre suivant.

**CHAPITRE III**

**ANALYSE DES BESOINS ET**

**CONCEPTION**

## **I. Introduction**

Après avoir défini dans les chapitres précédents respectivement l'état de l'art sur système d'intégration ainsi que la recherche d'informations. Nous passons à l'étape conception et modélisation afin de concevoir les schémas généraux qui permettent la modélisation du fonctionnement de l'application, cette dernière est basée sur l'intégration des sources de métadonnées par l'approche médiateur.

## **II. Le Cycle de vie d'un logiciel et le langage de modélisation UML**

### **II.1. Le cycle de vie d'un logiciel**

#### **II.1.1. Définition**

Le cycle de vie d'un logiciel (en anglais software lifecycle), désigne toutes les étapes du développement d'un logiciel, de sa conception à sa disparition. L'objectif d'un tel découpage est de permettre de définir des jalons intermédiaires permettant la validation du développement logiciel, c'est-à-dire la conformité du logiciel avec les besoins exprimés, et la vérification du processus de développement, c'est-à-dire l'adéquation des méthodes mises en œuvre [LAU, 2006].

#### **II.1.2. Modèle de cycle de vie en cascade**

Le modèle de cycle de vie en cascade a été mis au point dès 1966, puis formalisé aux alentours de 1970 [LAU, 2006].

Dans ce modèle le principe est très simple : chaque phase se termine à une date précise par la production de certains documents ou logiciels. Les résultats sont définis sur la base des interactions entre étapes, ils sont soumis à une revue approfondie et on ne passe à la phase suivante que s'ils sont jugés satisfaisants [LAU, 2006].

Le cycle de vie du logiciel comprend généralement au minimum les étapes suivantes :

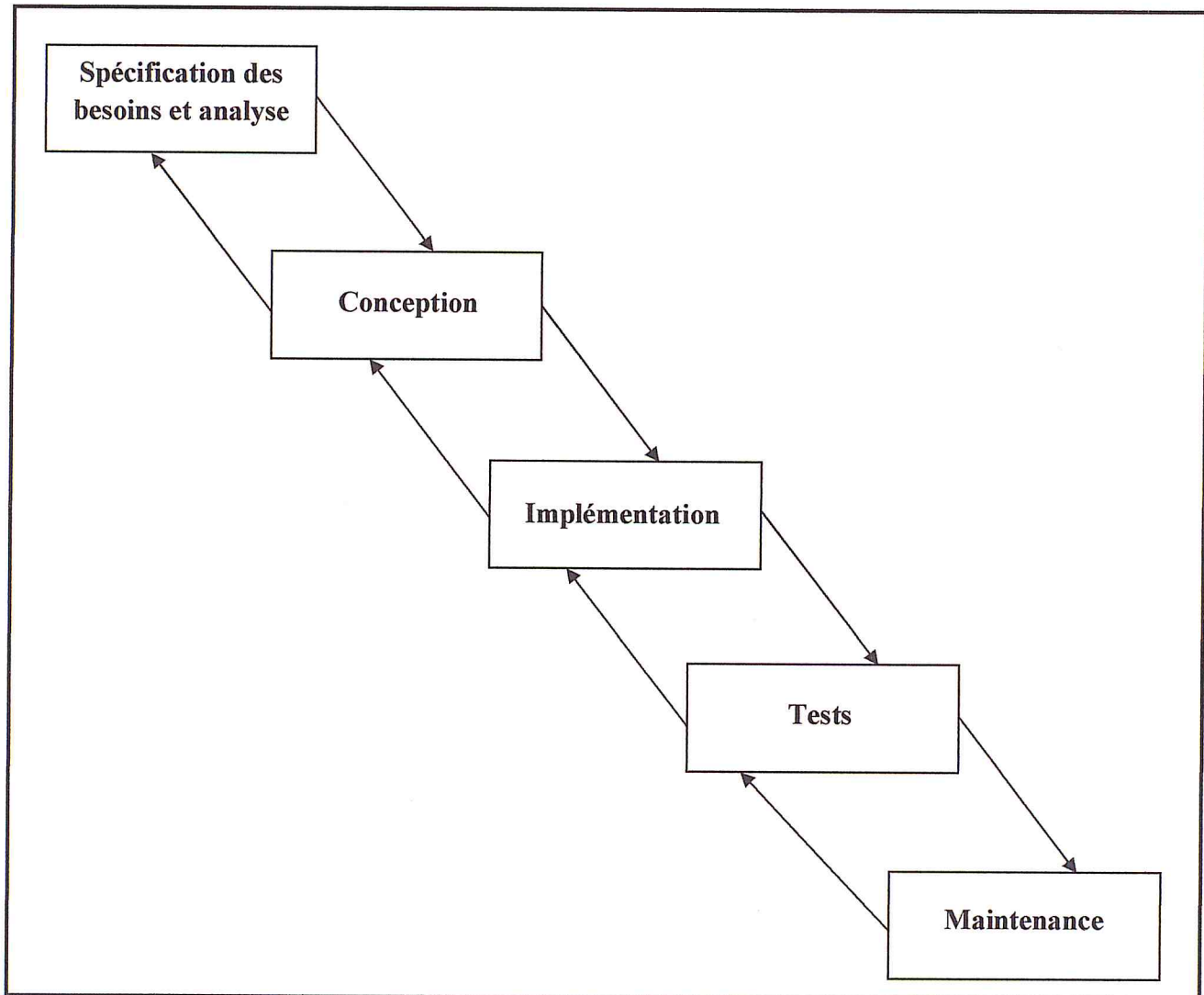


Figure III.1 : Le cycle de vie de modèle en cascade.

### II.1.3. Les activités

#### 1. Spécification des besoins et analyse

##### ❖ Spécification des besoins

L'expression des besoins comme son nom l'indique, permet de définir les différents besoins:

- inventorier les **besoins principaux** et fournir une liste de leurs fonctions

- recenser les **besoins fonctionnels** (du point de vue de l'utilisateur) qui conduisent à l'élaboration des modèles de cas d'utilisation
- appréhender les **besoins non fonctionnels** (technique) et livrer une liste des exigences.

Le modèle de cas d'utilisation présente le système du point de vue de l'utilisateur et représente sous forme de cas d'utilisation et d'acteur, les besoins du client.

#### ❖ **Analyse**

L'objectif de l'analyse est d'accéder à une compréhension des besoins et des exigences du client. Il s'agit de livrer des spécifications pour permettre de choisir la conception de la solution.

Un modèle d'analyse livre une spécification complète des besoins issus des cas d'utilisation et les structure sous une forme qui facilite la compréhension (scénarios), la préparation (définition de l'architecture), la modification et la maintenance du futur système.

Il s'écrit dans le langage des développeurs et peut être considéré comme une première ébauche du modèle de conception.

## **2. Conception**

La conception permet d'acquérir une compréhension approfondie des contraintes liées au langage de programmation, à l'utilisation des composants et au système d'exploitation.

Elle détermine les principales interfaces et les transcrit à l'aide d'une notation commune.

Elle constitue un point de départ à l'implémentation :

- elle décompose le travail d'implémentation en sous-système.
- elle crée une abstraction transparente de l'implémentation.

### **3. Implémentation**

L'implémentation est le résultat de la conception pour implémenter le système sous formes de composants, c'est-à-dire, de code source, de scripts, de binaires, d'exécutables et d'autres éléments du même type.

Les objectifs principaux de l'implémentation sont de planifier les intégrations des composants pour chaque itération, et de produire les classes et les sous-systèmes sous formes de codes sources.

### **4. Test**

Les tests permettent de vérifier des résultats de l'implémentation en testant la construction.

Pour mener à bien ces tests, il faut les planifier pour chaque itération, les implémenter en créant des cas de tests, effectuer ces tests et prendre en compte le résultat de chacun.

### **5. Maintenance**

Elle comprend toutes les actions correctives (maintenance corrective) et évolutives (maintenance évolutive) sur le logiciel.

## **II .2. Le langage UML**

### **II .2.1 Définition**

UML ou Unified Modeling Language, est un langage de modélisation qui est né au milieu des années 90 de la fusion de trois méthodes objets: OMT (Object Modeling Technique), BOOCH1 (GRADY BOOCH le concepteur de la méthode) et OOSE (Object Oriented Software Engineering). L'idée de cette fusion est partie du constat qu'à l'époque ils existaient plusieurs méthodes objets liées par un consensus autour d'idées communes: objets, classes, sous-systèmes etc. [ZEN, 2009].

C'est à partir de 1997 que l'OMG2 (Object Management Group) qui standardise les technologies de l'objet, s'est attachée à la définition d'un langage commun unique, utilisable par toutes les méthodes objets dans toutes les phases du cycle de vie et compatible avec les techniques de réalisation du moment. D'où la naissance d'UML.



UML offre des éléments pour décrire les différents aspects d'un système: les diagrammes.

Ses points forts sont les suivants :

- C'est un langage formel et normalisé : il permet un gain de précision, de stabilité et encourage l'utilisation d'outils.
- C'est un support de communication performant : il cadre l'analyse et facilite la compréhension de représentations abstraites complexes. De plus, son caractère polyvalent et sa souplesse en font un langage universel [ZEN, 2009].

UML 2.0 comporte ainsi treize types de diagrammes représentant autant de vues distinctes pour représenter des concepts particuliers du système d'information. Ils se répartissent en trois grands groupes : [LAU, 2006]

- **Diagrammes structurels ou diagrammes statiques (UML Structure)**
  - diagramme de classes (Class diagram)
  - diagramme d'objets (Object diagram)
  - diagramme de composants (Component diagram)
  - diagramme de déploiement (Deployment diagram)
  - diagramme de paquets (Package diagram)
  - diagramme de structures composites (Composite structure diagram)
- **Diagrammes comportementaux ou diagrammes dynamiques (UML Behavior)**
  - diagramme de cas d'utilisation (Use case diagram)
  - diagramme d'activités (Activity diagram)
  - diagramme d'états-transitions (State machine diagram)
- **Diagrammes d'interaction (Interaction diagram)**
  - diagramme de séquence (Sequence diagram)
  - diagramme de communication (Communication diagram)
  - diagramme global d'interaction (Interaction overview diagram)
  - diagramme de temps (Timing diagram)

### III. Spécification des besoins

#### III.1. Recueil des Besoins Fonctionnels

##### III.1.1. Identification des Acteurs

La première étape de cette phase est d'énumérer les Acteurs susceptibles d'interagir avec le système.

#### Définition

Un **Acteur** représente l'abstraction d'un rôle joué par des entités externes (utilisateur, dispositif matériel ou autre système), qui interagissent directement avec le système étudié [LAU, 2006].

Le tableau ci-dessous identifie les acteurs de notre système et décrit la mission de chacun.

➤ Les Acteurs

Acteur	Désignation
Utilisateur	Un Utilisateur est tout ce qui utilise notre système.
Administrateur	L'administrateur est un acteur principal dans notre Système

Tableau III.1 : Les acteurs du système.

➤ Description des besoins fonctionnels des acteurs

Acteur	Description des besoins fonctionnels
Utilisateur	L'application doit permettre aux utilisateurs de : <ul style="list-style-type: none"><li>• Poser une requête pour accéder aux données pertinentes de différentes sources de notre système.</li></ul>
Administrateur	L'application doit permettre aux administrateurs de : <ul style="list-style-type: none"><li>• S'authentifier : L'administrateur doit s'authentifier avec un nom et un mot de passe</li></ul> L'application doit permettre à l'administrateur de : <ul style="list-style-type: none"><li>• Créer des vues sur les sources.</li><li>• Créer une vue sur le schéma globale.</li><li>• Ajouter les métadonnées.</li></ul>

	<ul style="list-style-type: none"><li>• Supprimer des métadonnées.</li><li>• modifier les métadonnées.</li></ul> L'administrateur peut jouer le rôle d'un utilisateur.
--	--

Tableau III.2: Description des besoins fonctionnels.

### III.1.2. Identification des cas d'utilisation

#### III .1.2.1. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système. C'est le premier diagramme du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre [LAU, 2006].

Un cas d'utilisation (use case) représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier. Un cas d'utilisation modélise un service rendu par le système. Il exprime les interactions acteurs/système et apporte une valeur ajoutée « notable » à l'acteur concerné [ZEN, 2009].

**Remarque :** Les descriptions détaillées vont être organisées dans des tableaux de la façon suivante :

Élément	Signification
Cas d'utilisation	Le nom de cas d'utilisation
Acteur	L'acteur qui réalise ce cas d'utilisation
But	Le but de cas d'utilisation
Description	Une explication de cas d'utilisation
Pré condition	Les conditions qu'elles doivent être vérifiées afin de démarrer le cas d'utilisation
Post condition	Les résultats de cas d'utilisation

<b>Exception</b>	Les informations entrées par l'acteur
------------------	---------------------------------------

Tableau III.3: Modèle de représentation des descriptions détaillées des cas d'utilisations.

### 1. Diagramme de cas d'utilisation général

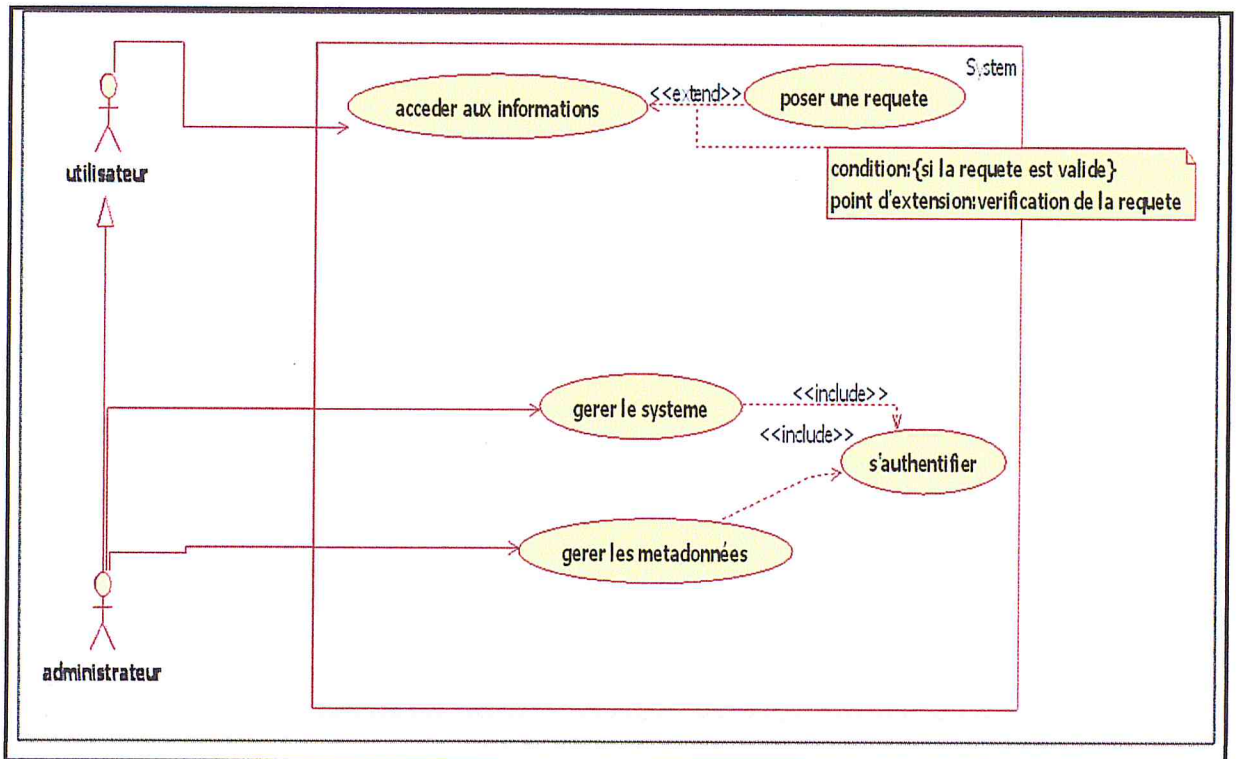


Figure III.2 : Diagramme de cas d'utilisation globale.

#### ➤ Authentification

<b>Cas d'utilisation</b>	Authentification.
<b>Acteur</b>	Administrateur.
<b>Objectifs</b>	Permettre à l'utilisateur de s'authentifier pour ouvrir sa session et accéder à ses privilèges.
<b>Description</b>	L'accès à l'espace approprié doit passer par le formulaire qui contient

	les renseignements adéquats.
<b>Pré condition</b>	L'administrateur doit fournir les informations convenables (le nom d'utilisateur et le mot de passe).
<b>Post condition</b>	L'administrateur peut effectuer les tâches qui lui sont permises.
<b>Exception</b>	Annulation ; si l'administrateur tape un nom d'utilisateur ou un mot de passe qui ne convient pas, le système affiche un message d'erreur.

Tableau III.4 : Description du cas d'utilisation « Authentification »

➤ **gérer les métadonnées**

<b>Cas d'utilisation</b>	Gérer les métadonnées.
<b>Acteur</b>	Administrateur.
<b>Objectifs</b>	Gérer les métadonnées (création, ajout, modification, suppression).
<b>Description</b>	La mise à jour (Ajouter, modifier et supprimer) des métadonnées doit passer par un formulaire.
<b>Pré condition</b>	L'administrateur doit s'authentifier.
<b>Post condition</b>	Mise à jour de la base de métadonnées.
<b>Exception</b>	Annulation ; si l'administrateur Ajoute les métadonnées qui existe déjà, Le système affiche un message « métadonnée existe déjà ».

Tableau III.5 : Description du cas d'utilisation « gérer les métadonnées »

2. Diagramme de cas d'utilisation détaillé pour l'administrateur

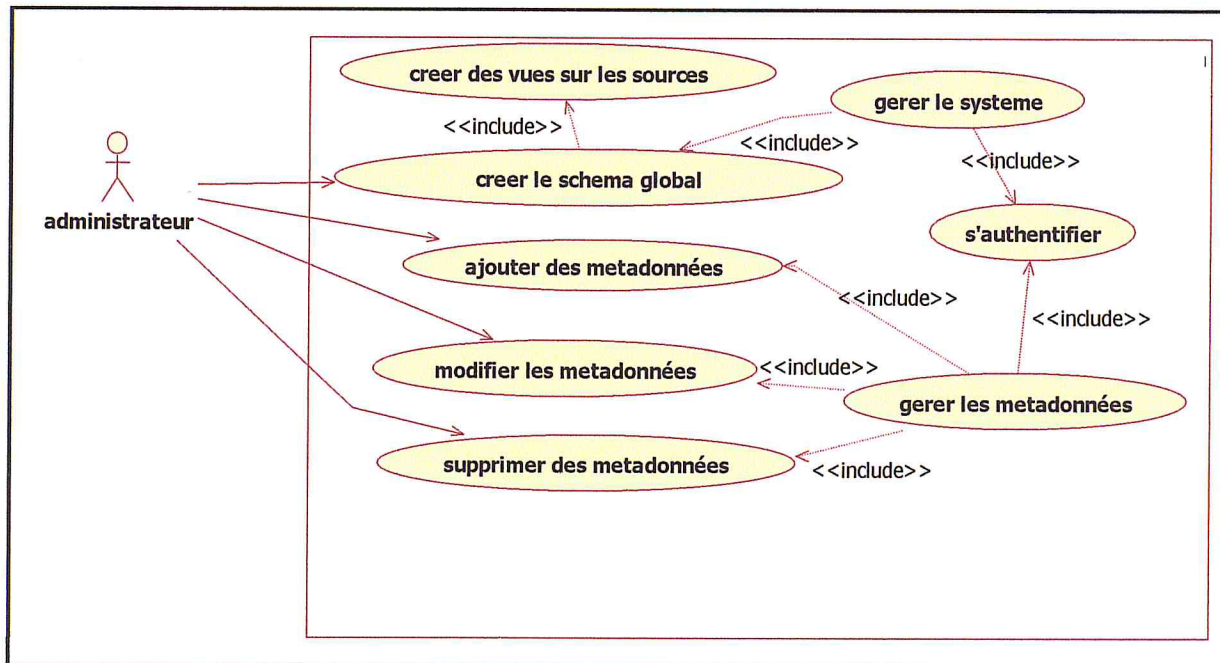


Figure III.3 : Diagramme de cas d'utilisation détaillé pour l'administrateur.

➤ créer des vues sur les sources

<b>Cas d'utilisation</b>	créer des vues sur les sources.
<b>Acteur</b>	Administrateur.
<b>Objectifs</b>	Pour pouvoir décomposer la requête de l'utilisateur en suivant les vues sur les sources.
<b>Description</b>	L'administrateur doit créer des vues sur les sources au niveau de médiateur.
<b>Pré condition</b>	L'administrateur doit s'authentifier.
<b>Post condition</b>	Le médiateur a des vues sur les sources de données.
<b>Exception</b>	Pas d'exception.

Tableau III.6 : Description du cas d'utilisation « créer des vues sur les sources »

➤ **créer une vue sur le schéma global**

<b>Cas d'utilisation</b>	créer un schéma global.
<b>Acteur</b>	Administrateur.
<b>Objectifs</b>	Pour fournir a l'utilisateur une vue homogène sur le système .
<b>Description</b>	L'administrateur doit créer une vue uniforme qui fusionne toute les vues sur les schémas locaux.
<b>Pré condition</b>	L'administrateur doit s'authentifier.
<b>Post condition</b>	Le médiateur a des vues sur les sources de données.
<b>Exception</b>	Pas d'exception.

Tableau III.7 : Description du cas d'utilisation « créer une vue sur le schéma global »

#### **IV. Analyse de système**

Dans la phase d'analyse nous présenterons les diagrammes d'activité et les diagrammes de séquences :

##### **IV .1 Diagramme d'activités**

Le diagramme d'activités n'est autre que la transcription dans UML de la représentation du processus telle qu'elle a été élaborée lors du travail qui a préparé la modélisation : il montre l'enchaînement des activités qui concourent au processus [LAU, 2006].

##### **1. Diagrammes d'activités pour l'utilisateur**

Le cas d'utilisation « Traiter le requete» est représenté par le diagramme d'activité suivant :

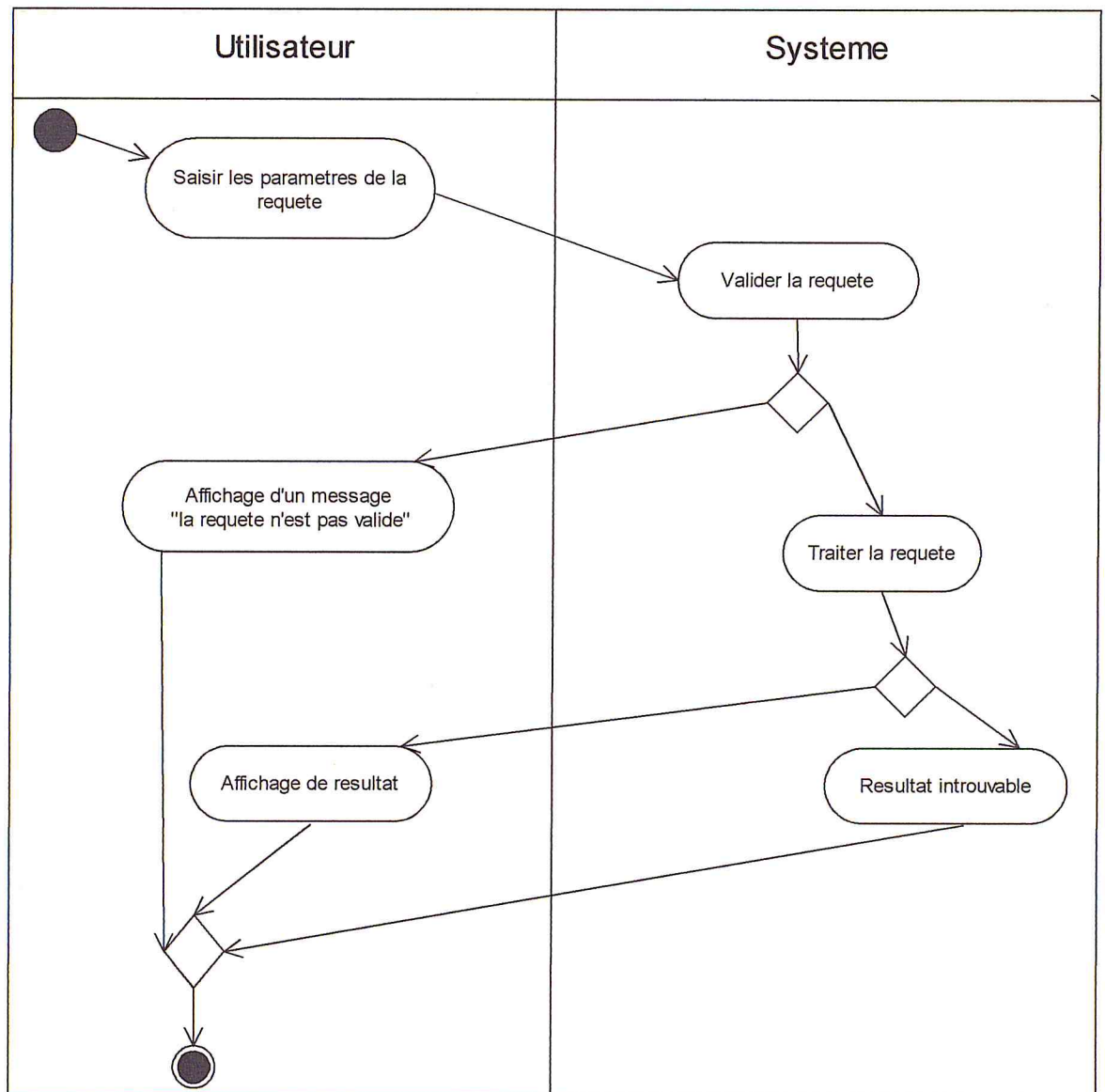


Figure III.4 : diagramme d'activité du cas d'utilisation « Traiter la requête ».

## 2. Diagrammes d'activités pour le médiateur

Le Médiateur doit :

- Traiter la requête de l'utilisateur.
- Décomposer la requête en sous requêtes.
- Recomposer les résultats des sous requêtes.



- Etablir les correspondances entre le schéma global et les schémas locaux.
- Le processus « **Décomposer la requête** » est représenté par le diagramme d'activité suivant :

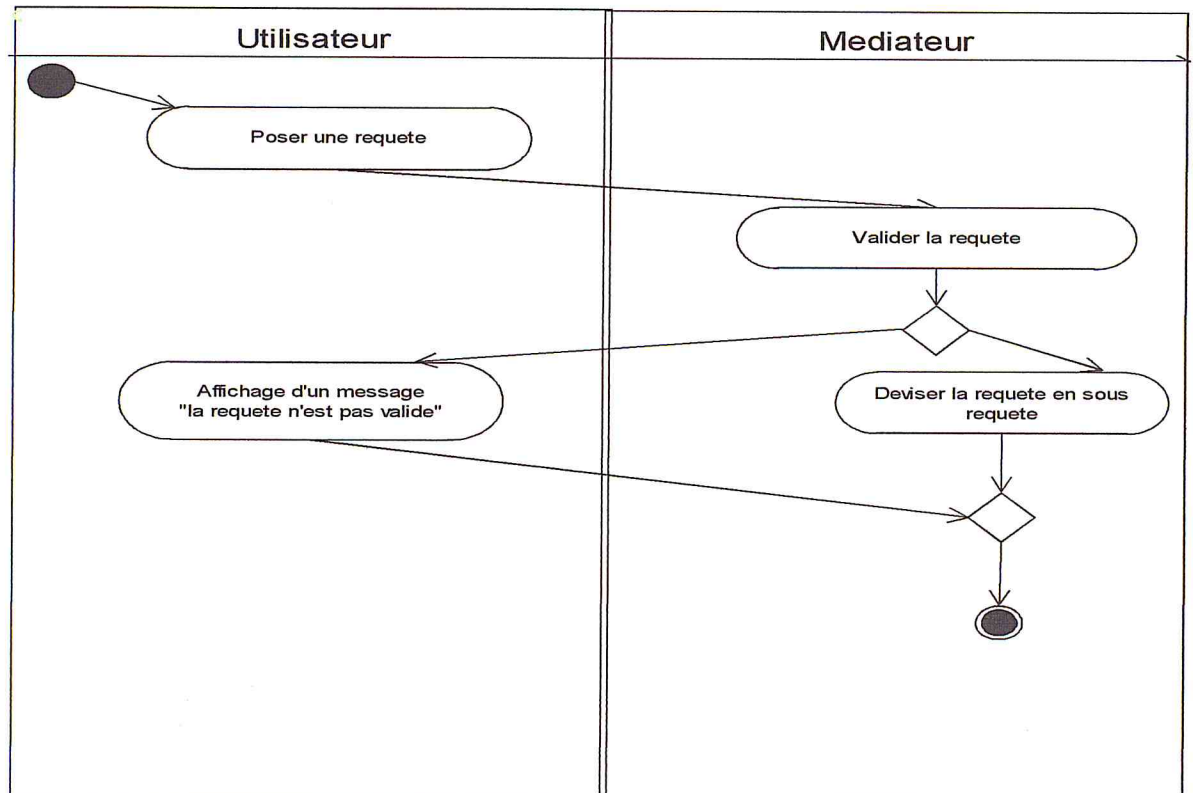


Figure III.5 : diagramme d'activité du processus « Décomposer la requête ».

- Le processus « **Recomposer les resultats** » est représenté par le diagramme d'activité suivant :

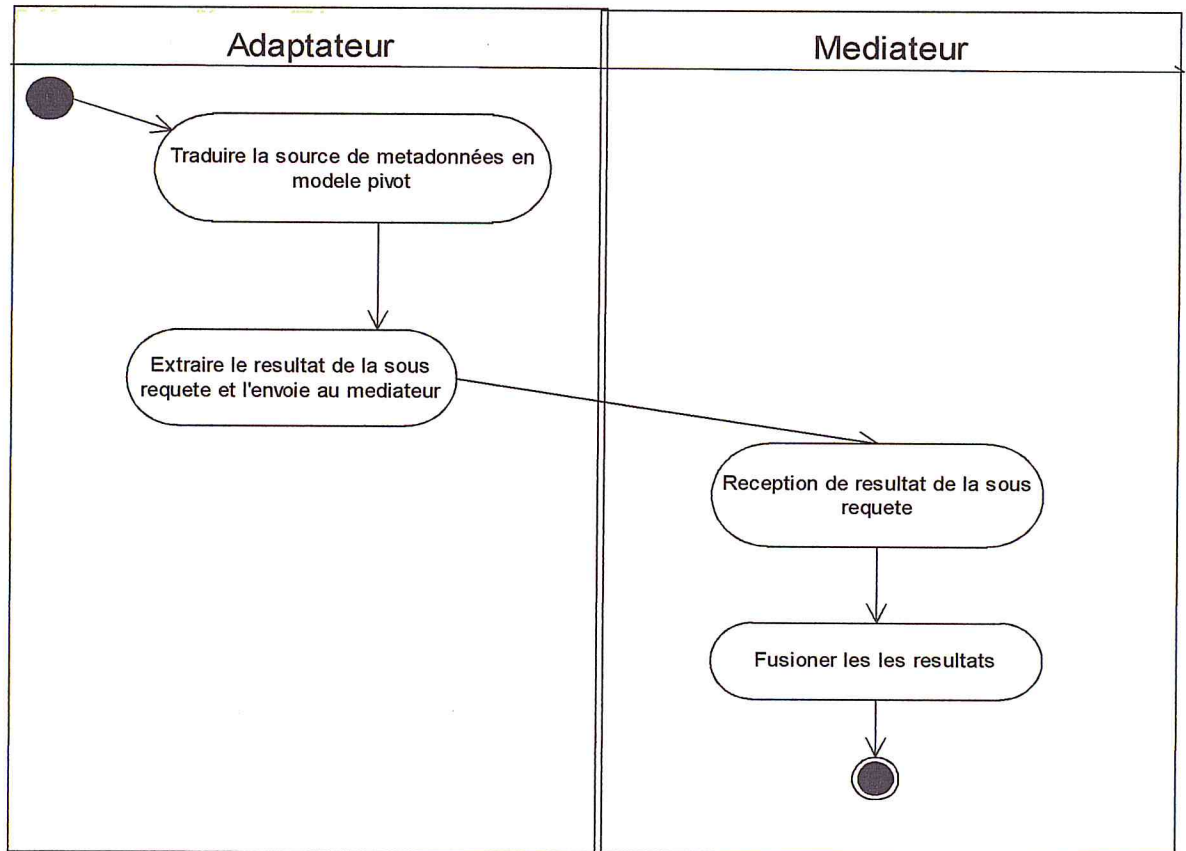


Figure III.6 : diagramme d'activité du processus « Recomposer les résultats ».

### 3. Diagrammes d'activités pour l'adaptateur

L'adaptateur doit:

- Traduire la source en modèle pivot.
- Extraire le résultat de la sous requête.
- envoyer le résultat en modèle pivot au médiateur.

### 4. Diagrammes d'activités pour l'administrateur

Le cas d'utilisation « **Authentification** » est représenté par le diagramme d'activité suivant :

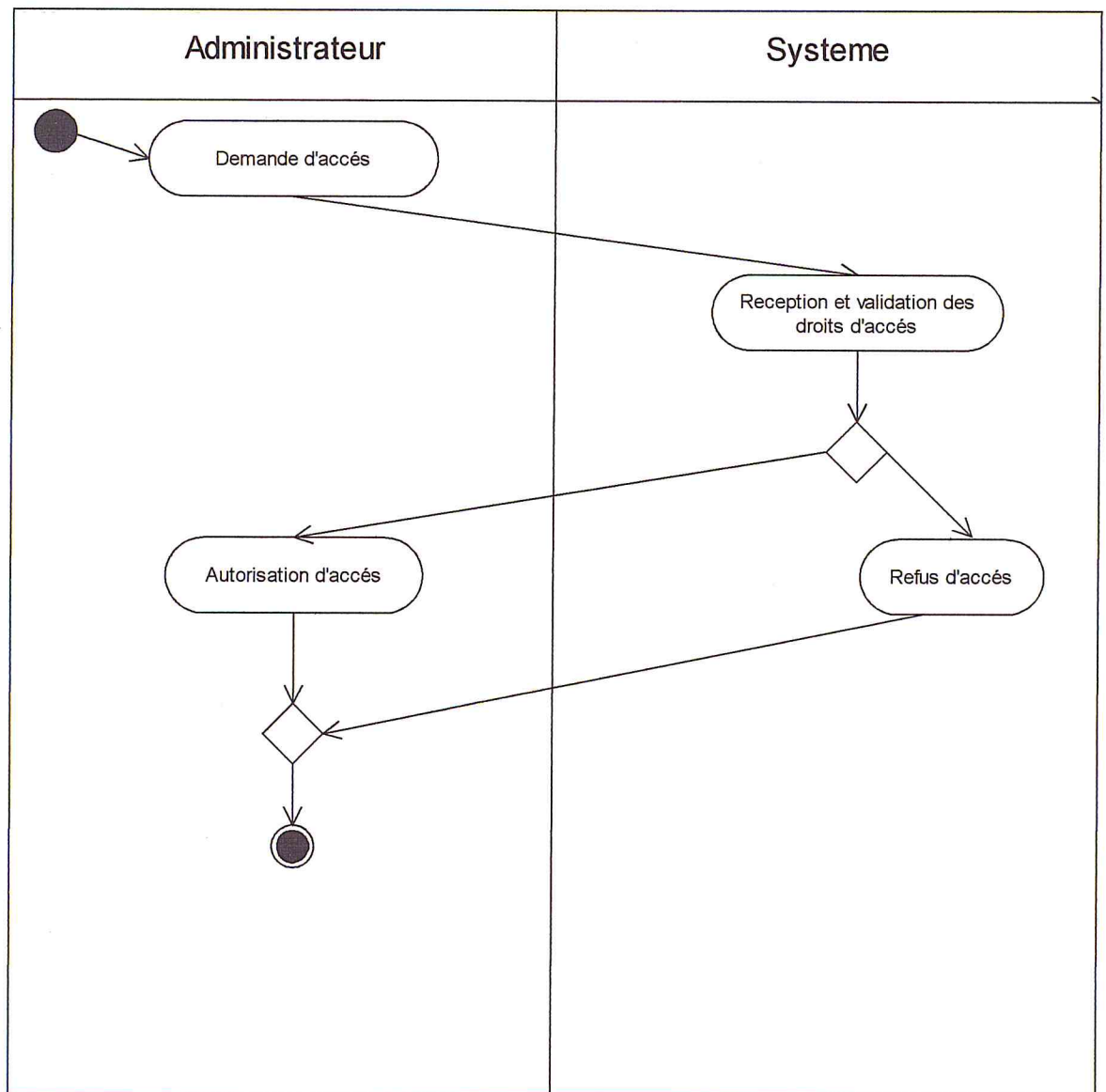


Figure III.7 : diagramme d'activité du cas d'utilisation « authentification ».

Le cas d'utilisation « **créer le schema global** » est representé par le diagramme d'activité suivant :

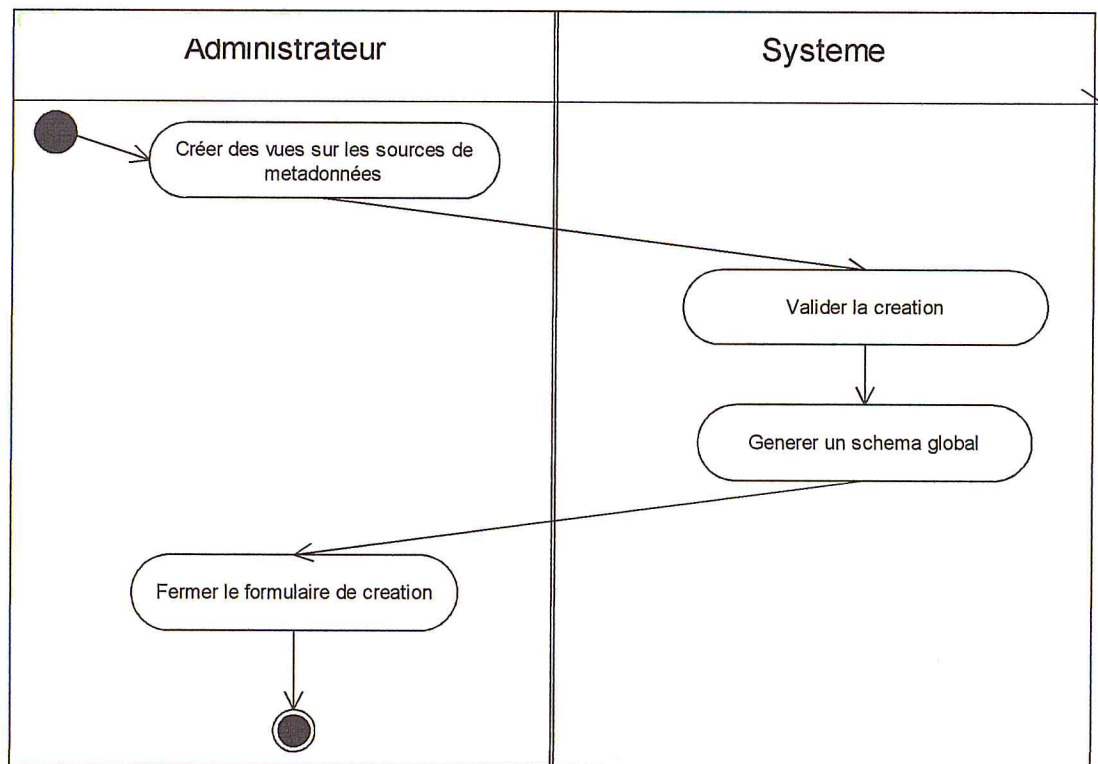


Figure III.8 : diagramme d'activité du cas d'utilisation « créer le schéma global ».

Le cas d'utilisation « **Gerer les metadonnées** » est représenté par le diagramme d'activité suivant :

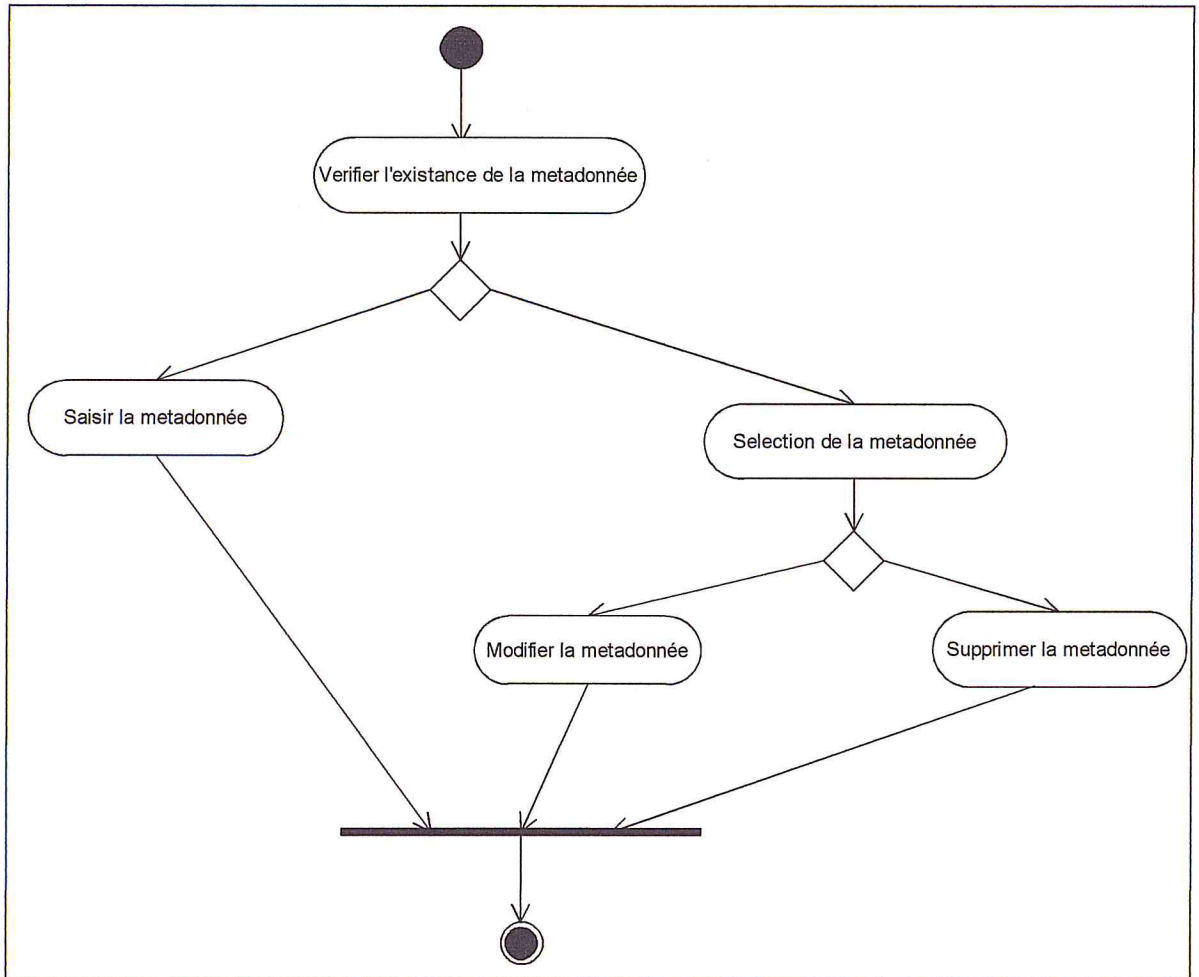


Figure III.9 : diagramme d'activité du cas d'utilisation « Gérer les métadonnées ».

Le cas d'utilisation « **Ajouter une metadonnée** » est représenté par le diagramme d'activité suivant :

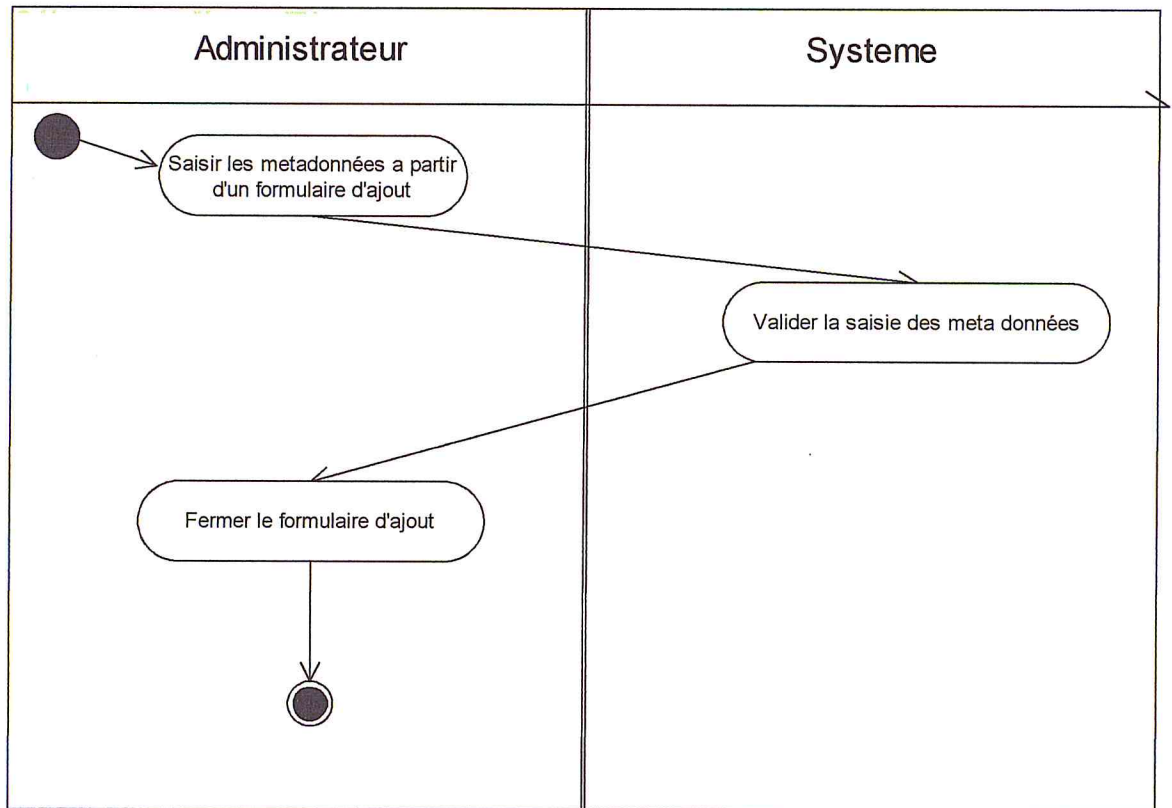


Figure III.10 : diagramme d'activité du cas d'utilisation « Ajouter une métadonnée ».

Le cas d'utilisation « **modifier une metadonnée** » est représenté par le diagramme d'activité suivant :

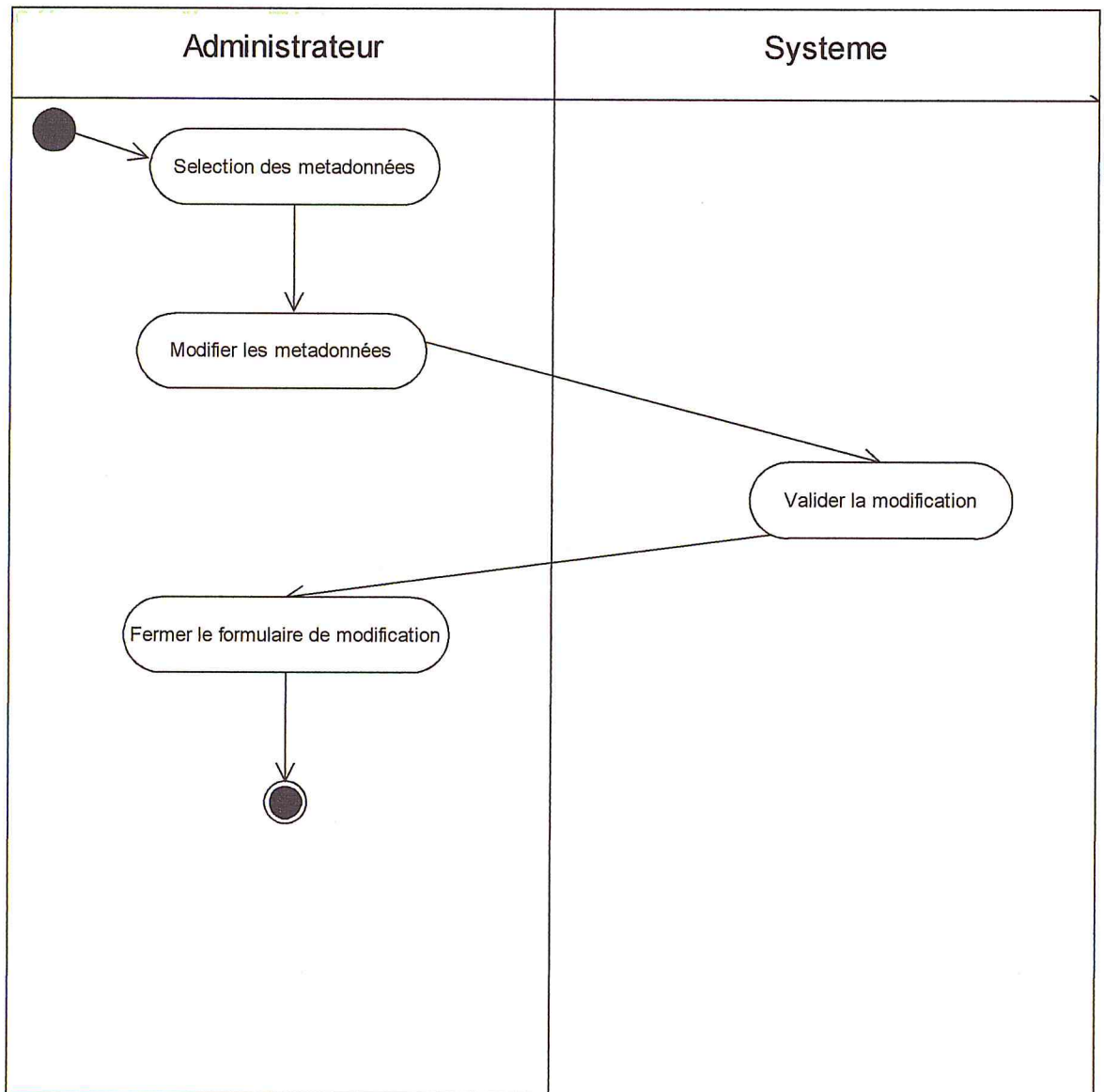


Figure III.11 : diagramme d'activité du cas d'utilisation « modifier les métadonnées ».

Le cas d'utilisation « **Supprimer une metadonnée** » est représenté par le diagramme d'activité suivant :

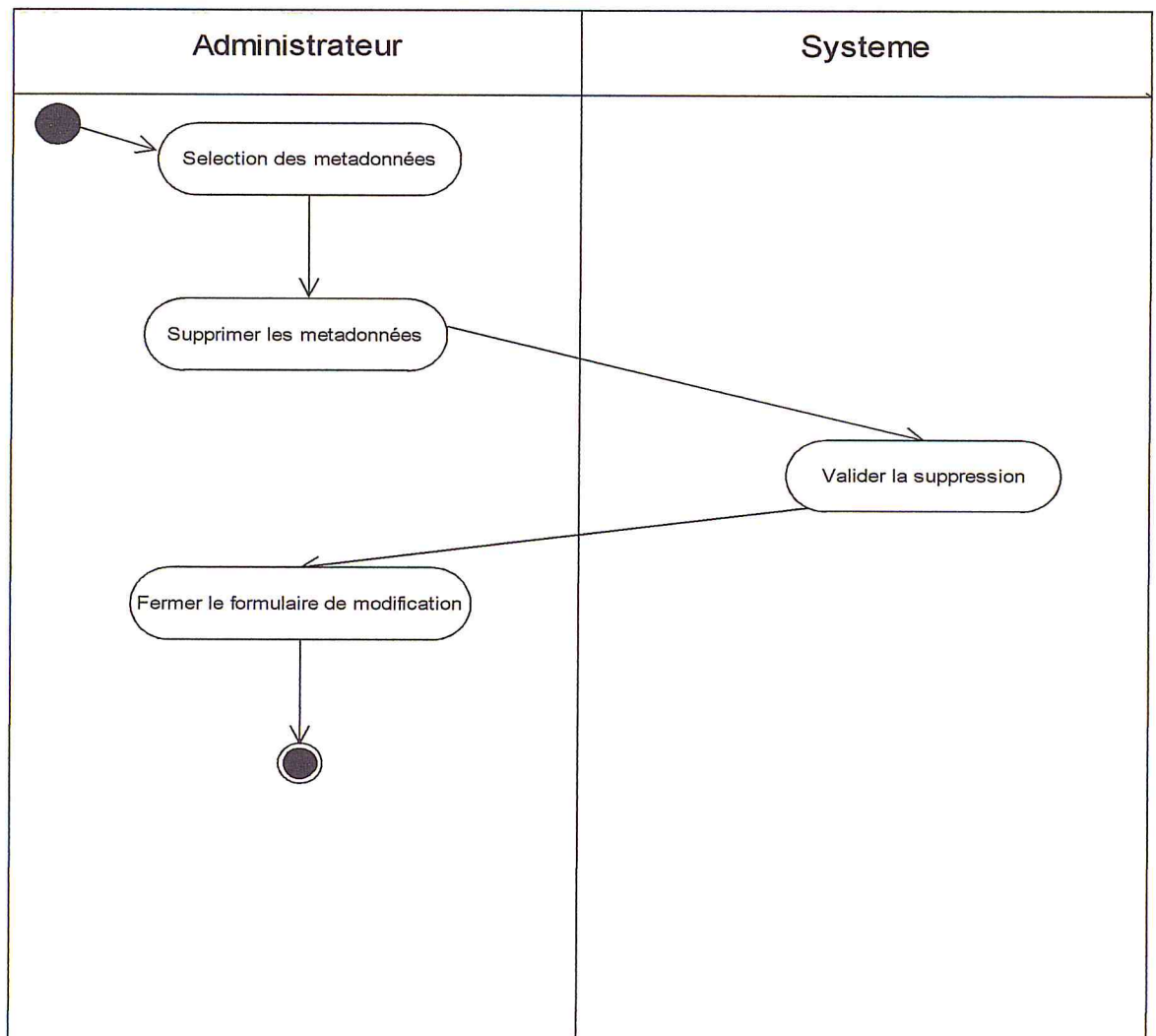


Figure III.12 : diagramme d'activité du cas d'utilisation « supprimer une métadonnée ».

Dans ce qui suit nous allons présenter les diagrammes de séquence afin de formaliser les scénarios des cas d'utilisation vus précédemment. Nous allons voir le système comme un ensemble d'objets en interaction.

#### IV .2 Diagramme de séquence

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre. On peut représenter les mêmes opérations par un



graphe dont les nœuds sont des objets et les arcs (numérotés selon la chronologie) les échanges entre objets [LAU, 2006].

#### IV .2 .1 Présentation des scénarios et diagrammes de séquence du Système

##### 1. Authentification

➤ **Scénario de l'authentification (cas normal)**

1. L'administrateur entre au Système.
2. Le système affiche la page d'authentification.
3. L'administrateur saisit le nom d'utilisateur et le mot de passe.
4. Le système vérifie le nom d'utilisateur et le mot de passe.
5. Le système affiche la page d'accueil.

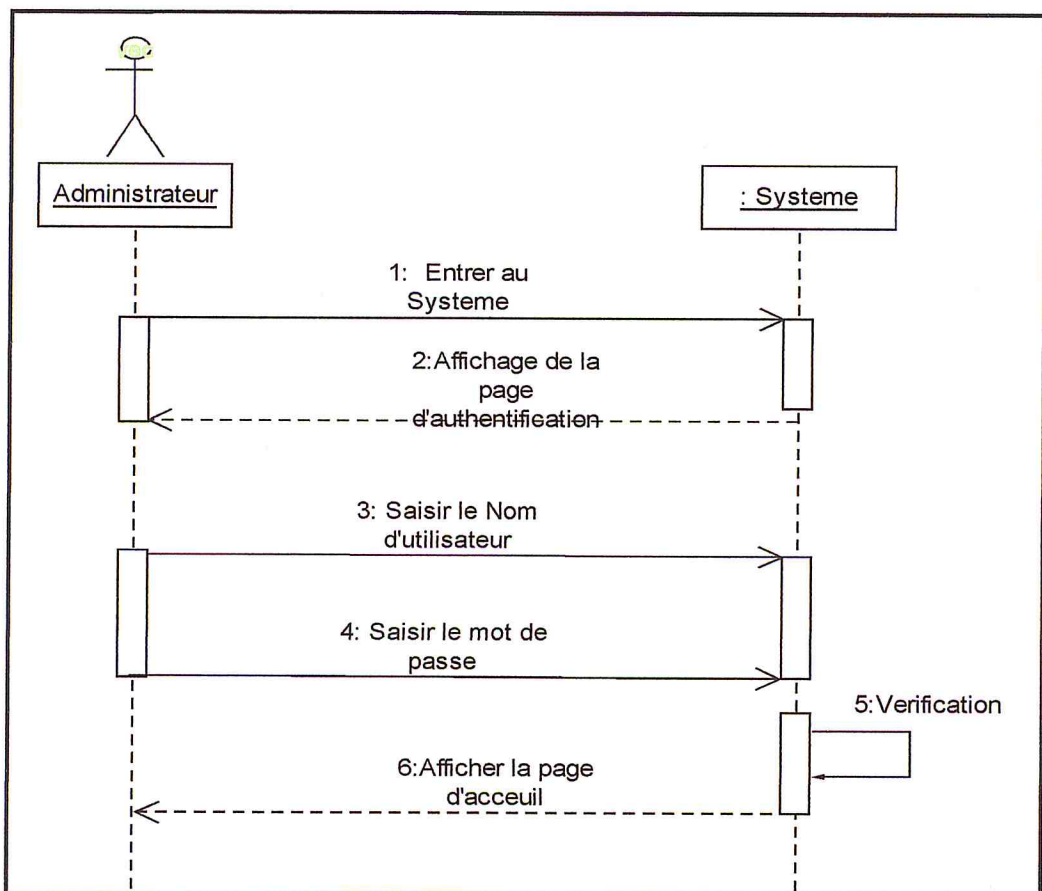


Figure III.13 : Diagramme de séquence de processus d'authentification (cas normal)

➤ **Scenario de l'authentification (cas d'erreur)**

1. L'administrateur entre au Système.
2. Le système affiche la page d'authentification.
3. L'administrateur saisit le nom d'utilisateur et le mot de passe.
4. Le système vérifie le nom d'utilisateur et le mot de passe.
5. Le système affiche «le nom d'utilisateur et le mot de passe n'est pas valide».

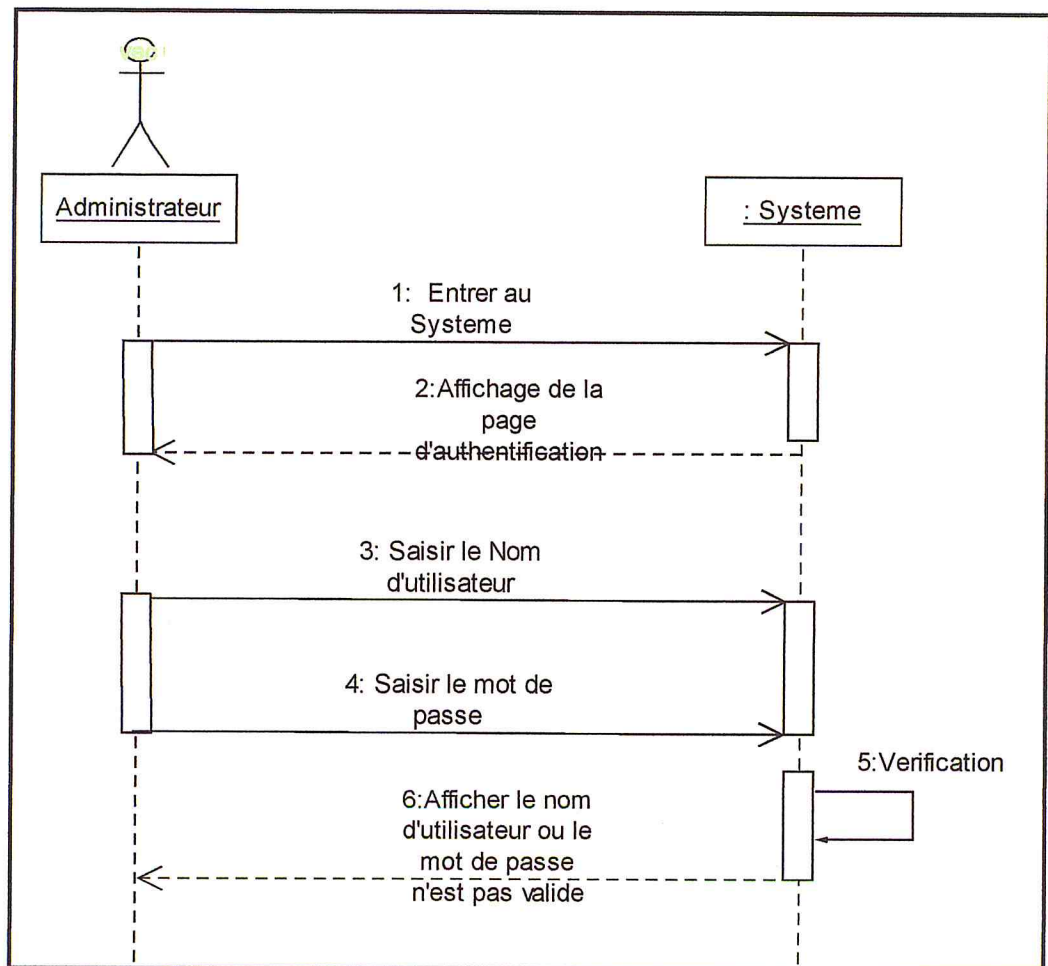


Figure III.14 : Diagramme de séquence de processus d'authentification (cas d'erreur)

**2 .Traiter la requête**

➤ **Scenario de « Traiter la requête » d'une manière détaillée.**

- 1 : Le système affiche la page d'accueil.
- 2 : Choisir « accéder aux données ».
- 3 : Renvoyer la page.
- 4 : Saisir la requête.
- 5 : Vérification de la requête.
- 6 : Envoie la requête au médiateur.
- 7 : Décomposer la requête en sous requête.
- 8 : Envoyer une sous requête à l'adaptateur XML.
- 9 : Envoyer une sous requête a l'adaptateur RDF.
- 10 : Envoyer une sous requête a l'adaptateur RuleML.
- ✕ 11 : Traduire la source RDF en XML.
- ✕ 12 : Traduire la source RuleML en XML.
- ✕ 13 : Demande la réponse de la sous requête envoyée par l'adaptateur XML.
- ✕ 14 : Demande la réponse de la sous requête envoyée par l'adaptateur RDF.
- ✕ 15 : Demande la réponse de la sous requête envoyée par l'adaptateur RuleML.
- ✕ 16 : Renvoyer la réponse de la requête envoyée par l'adaptateur XML.
- ✕ 17 : Renvoyer la réponse de la requête envoyée par l'adaptateur RDF.
- ✕ 18 : Renvoyer la réponse de la requête envoyée par l'adaptateur RuleML.
- ✕ 19 : Renvoyer le résultat de la sous requête par l'adaptateur XML.
- ✕ 20 : Renvoyer le résultat de la sous requête par l'adaptateur RDF.
- 21 : Renvoyer le résultat de la sous requête par l'adaptateur RuleML.
- 22 : Recomposer les résultats.
- 23 : Renvoyer le résultat final.
- 24 : Affichage du résultat.

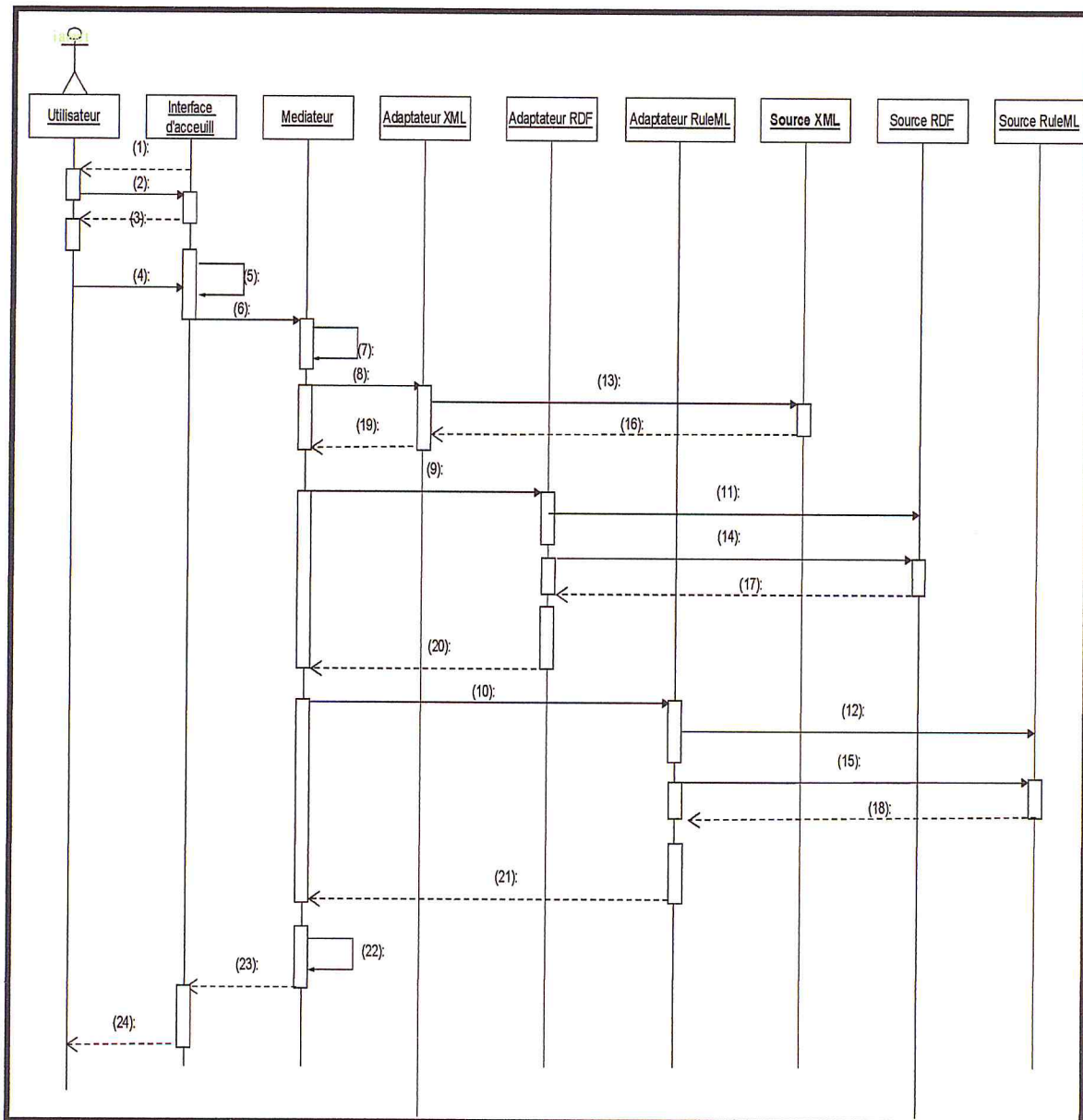


Figure III.15 : Diagramme de séquence de processus « Traiter la requête » (Détailé)

### 3. ajouter une métadonnée

#### ➤ Scenario de processus « ajouter une métadonnée »

1 : Demande d'afficher la page « Administrateur ».

2 : Affichage de la page « Administrateur ».

- 3 : Choisir : Ajouter les métadonnées.
- 4 : Afficher la page d'ajout de métadonnées.
- 5 : Remplir le formulaire d'ajout.
- 6 : Vérifier l'existence des métadonnées.
- 7 : Métadonnées non existées.
- 8 : Valider l'ajout des métadonnées.

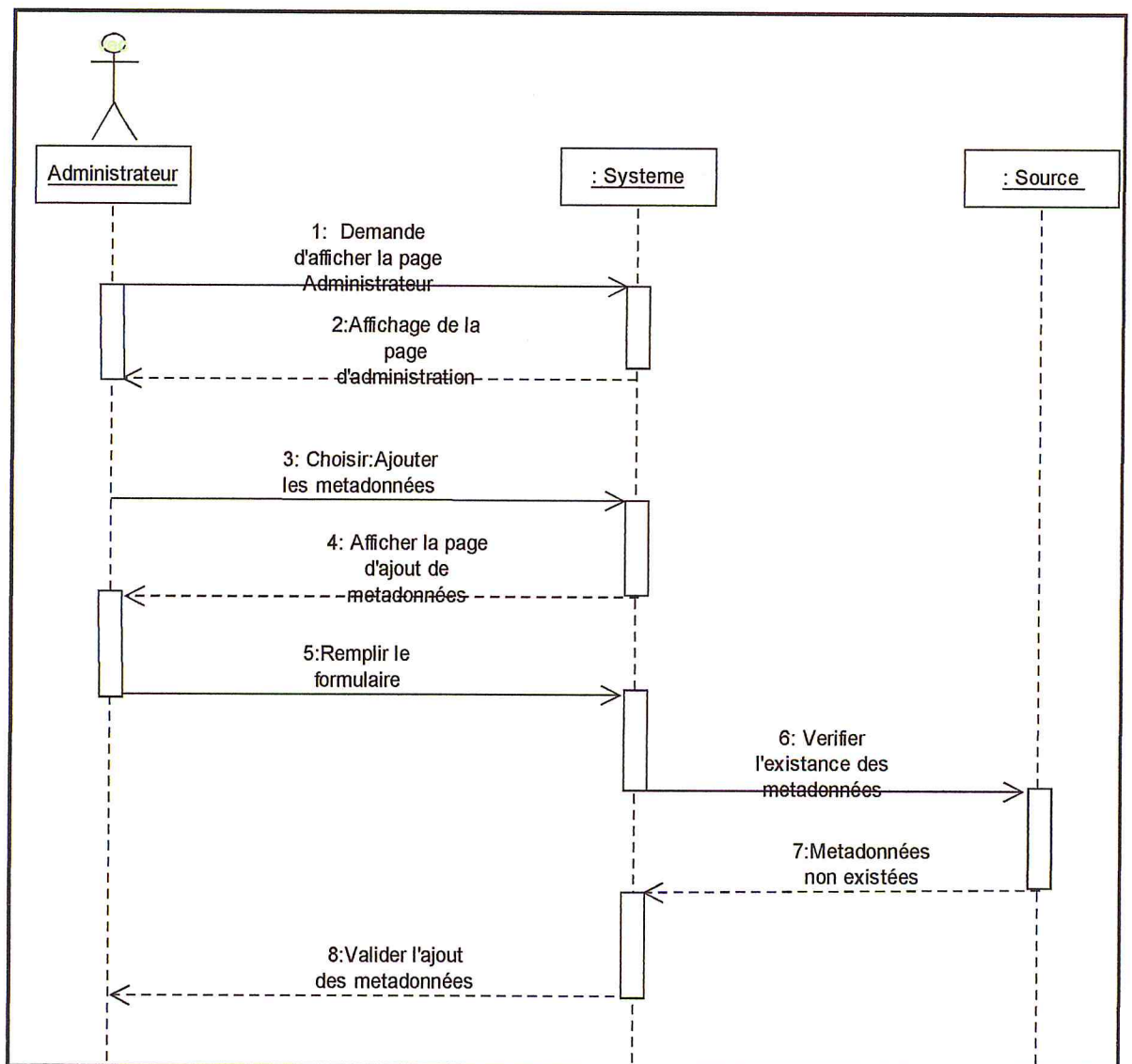


Figure III.16 : Diagramme de séquence de processus « Ajouter une métadonnée ».

#### **4. Modifier les métadonnées**

➤ **Scenario de processus « Modifier les métadonnées »**

- 1 : Demande d'afficher la page « Administrateur ».
- 2 : Affichage de la page « Administrateur ».
- 3 : Choisir : Modifier les métadonnées.
- 4 : Chercher les données.
- 5 : Renvoyer le résultat.
- 6 : Afficher la liste des données.
- 7 : Sélectionner une donnée a modifié leurs métadonnées.
- 8 : Afficher le formulaire de modification.
- 9 : Remplir les nouvelles métadonnées.
- 10 : Envoyer les nouvelles métadonnées.
- 11 : Valider la saisie.
- 12 : Métadonnées modifiées avec succès.

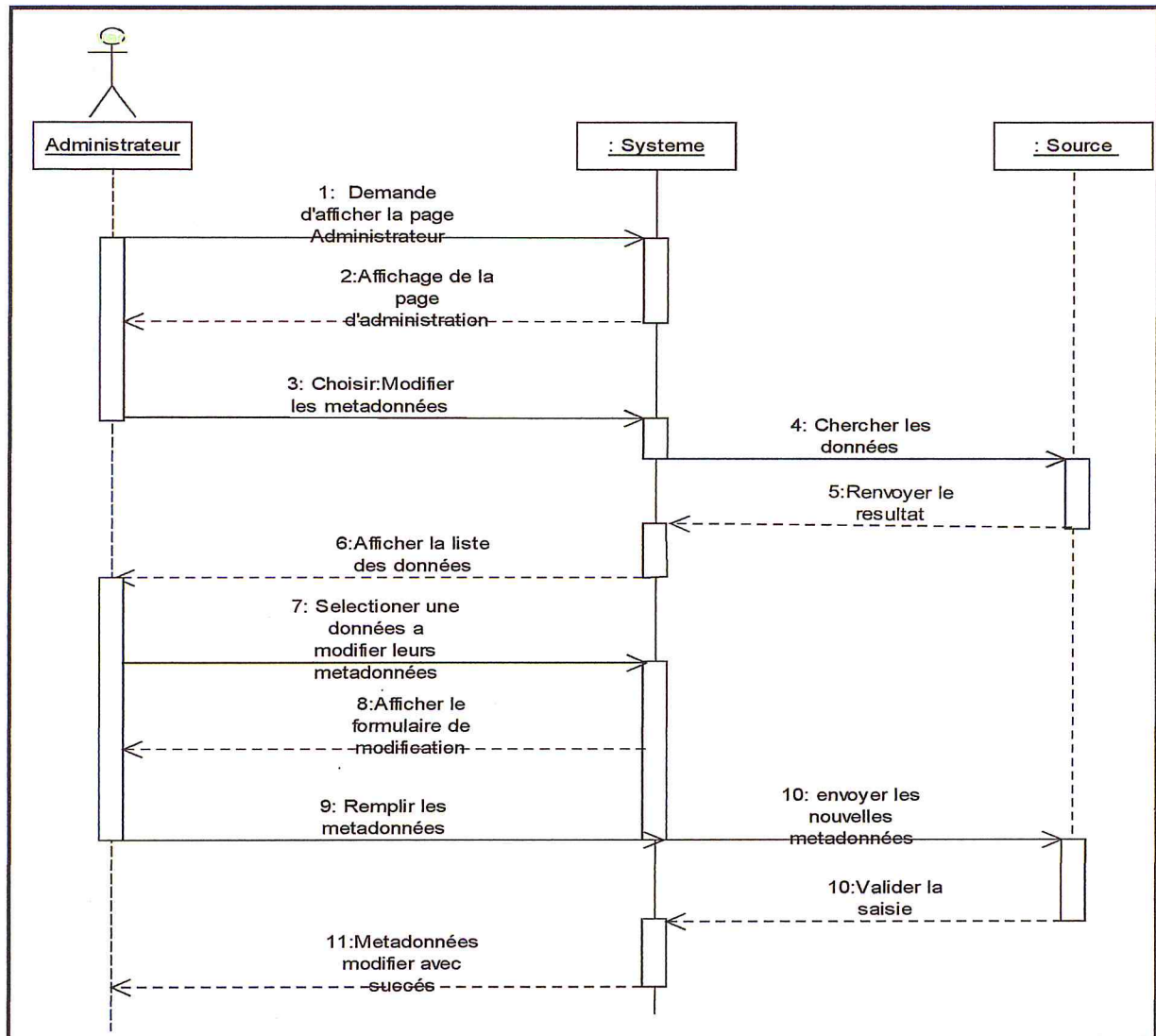


Figure III.17 : Diagramme de séquence de processus « Modifier les métadonnées »

## 5. Supprimer une métadonnée

### ➤ Scenario de processus « Supprimer une métadonnée »

- 1 : Demande d'afficher la page « Administrateur ».
- 2 : Affichage de la page « Administrateur ».
- 3 : Choisir : Supprimer une métadonnée.
- 4 : Chercher les données.

- 5 : Renvoyer le résultat.
- 6 : Afficher la liste des données.
- 7 : Sélectionner une donnée a supprimé leurs métadonnées.
- 8 : Afficher le formulaire de suppression.
- 9 : Supprimer les métadonnées.
- 10 : Envoyer la donnée sans leurs métadonnées.
- 11 : Valider la suppression.
- 12 : Métadonnées supprimées avec succès.

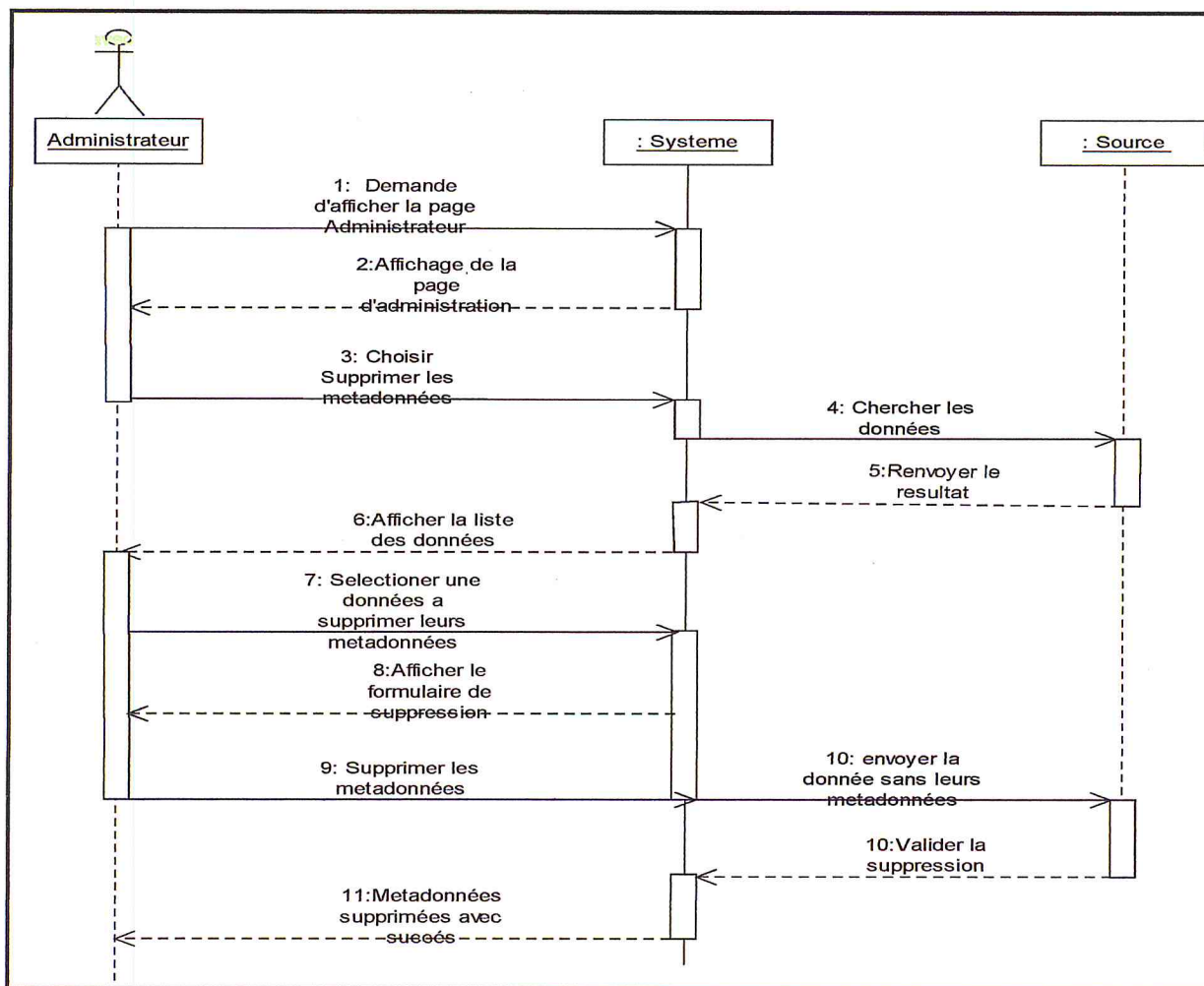


Figure III.18 : Diagramme de séquence de processus « Supprimer une métadonnée ».



## **V. Conception**

### **V.1. Identification des classes candidates**

Cette phase va préparer la **modélisation orientée objet** en aidant à trouver les classes principales du futur modèle statique d'analyse.

### **V.2. Diagramme de classes**

Le diagramme de classes est généralement considéré comme le plus important dans un développement orienté objet. Il représente l'architecture conceptuelle du système : il décrit les classes que le système utilise, ainsi que leurs liens, que ceux-ci représentent un emboîtement conceptuel (héritage) ou une relation organique (agrégation) [LAU, 2006].

#### **Le diagramme de classe de notre Système**

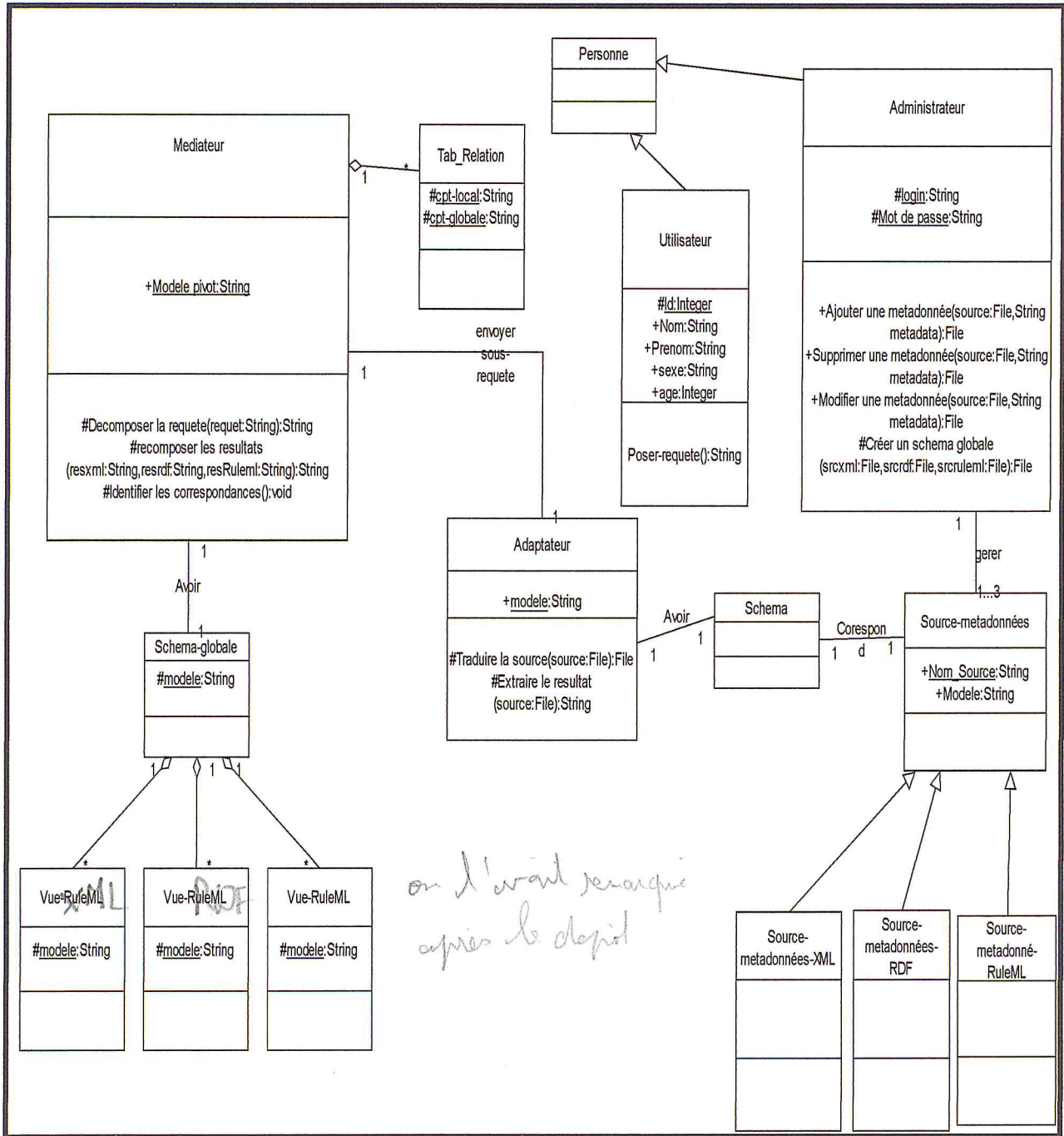


Figure III.19 : Diagramme de classe de notre système.

## VI. Architecture générale de notre système de médiation

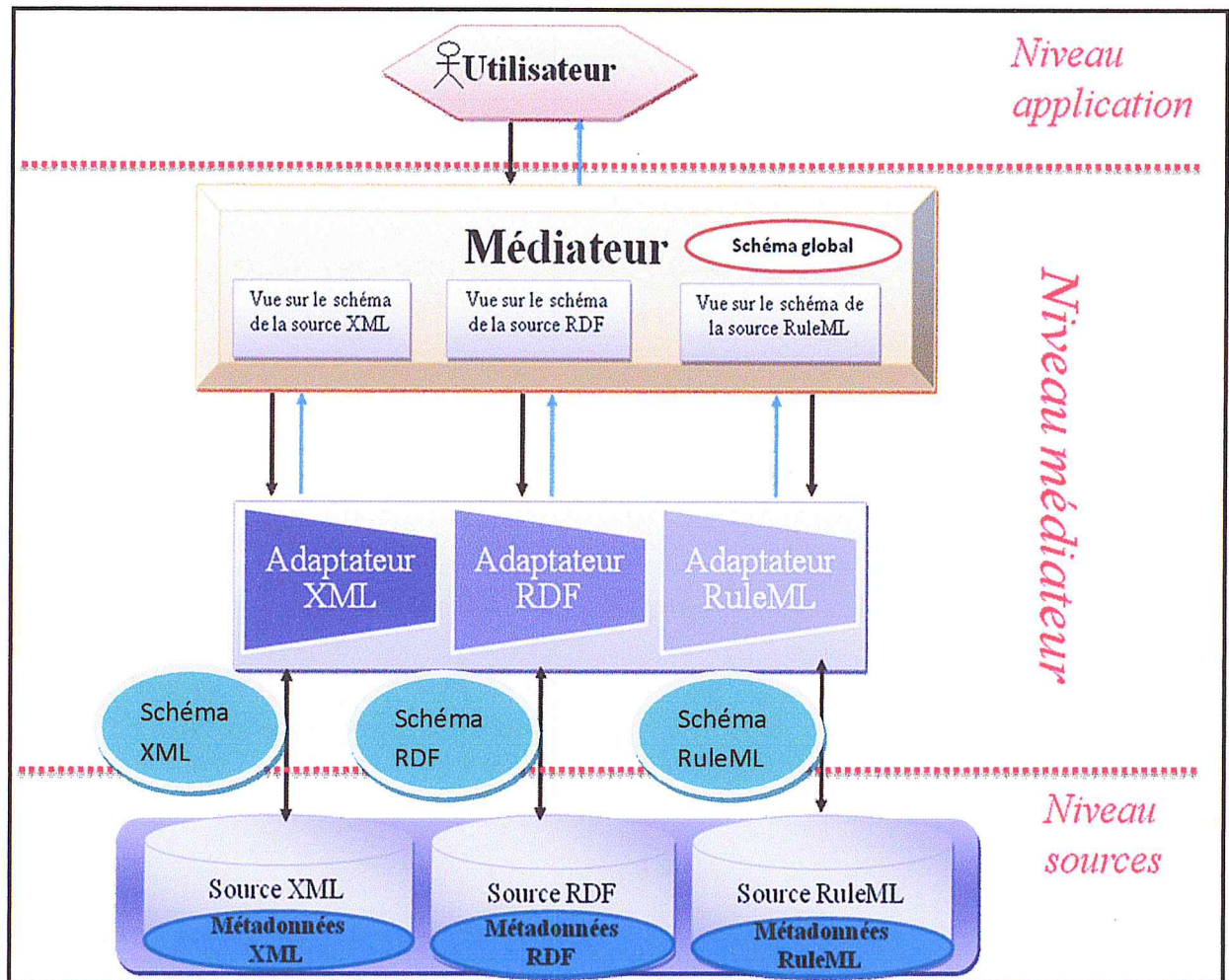


Figure III.20. Architecture de notre système d'intégration.

### VI.1 Description des couches de système

Notre système d'intégration a une architecture à 4 couches, de bas en haut commençons par :

#### VI.1.1 Couche Source de Données

Dans un processus d'intégration, les sources de métadonnées utilisées peuvent être de nature structurée, semi structurée ou bien non structurée. Dans notre approche les

données sont semi structurées du fait qu'on s'intéresse à des documents XML, RDF, RuleML.

Pour réaliser notre travail, nous avons choisi trois sources de métadonnées, chacune est un fragment du schéma global.

➤ **Une source de métadonnées RDF** : c'est une source de métadonnée semi structurée qui porte le nom «Examen Médical»

Le schéma suivant présente le contenu de cette source :

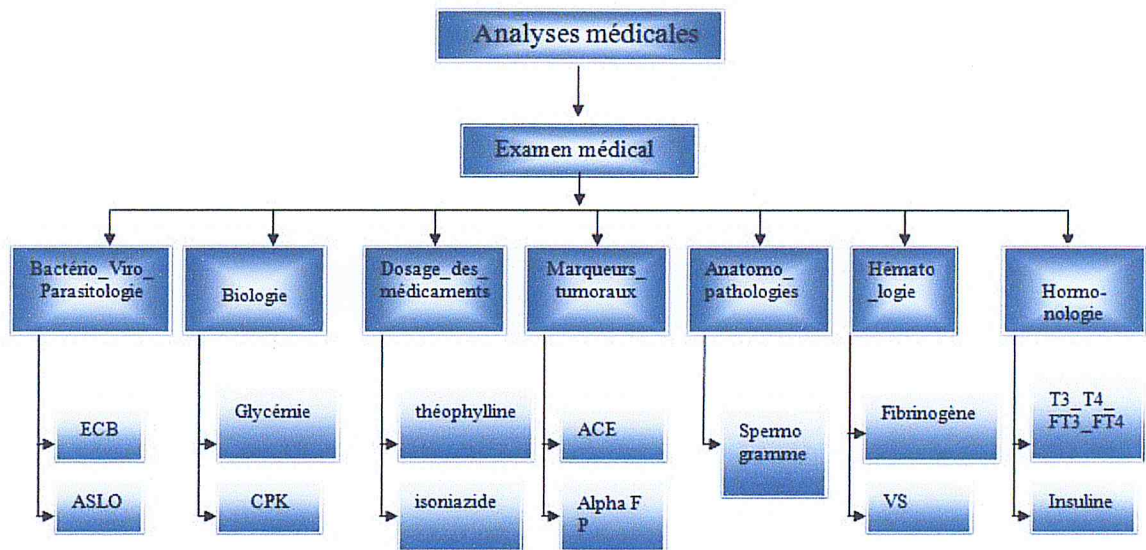


Figure III.21. Architecture de source RDF.

➤ **Une deuxième source de métadonnées XML** : c'est une source de métadonnées qui porte le nom «Test» Le schéma suivant présente le contenu de cette source :

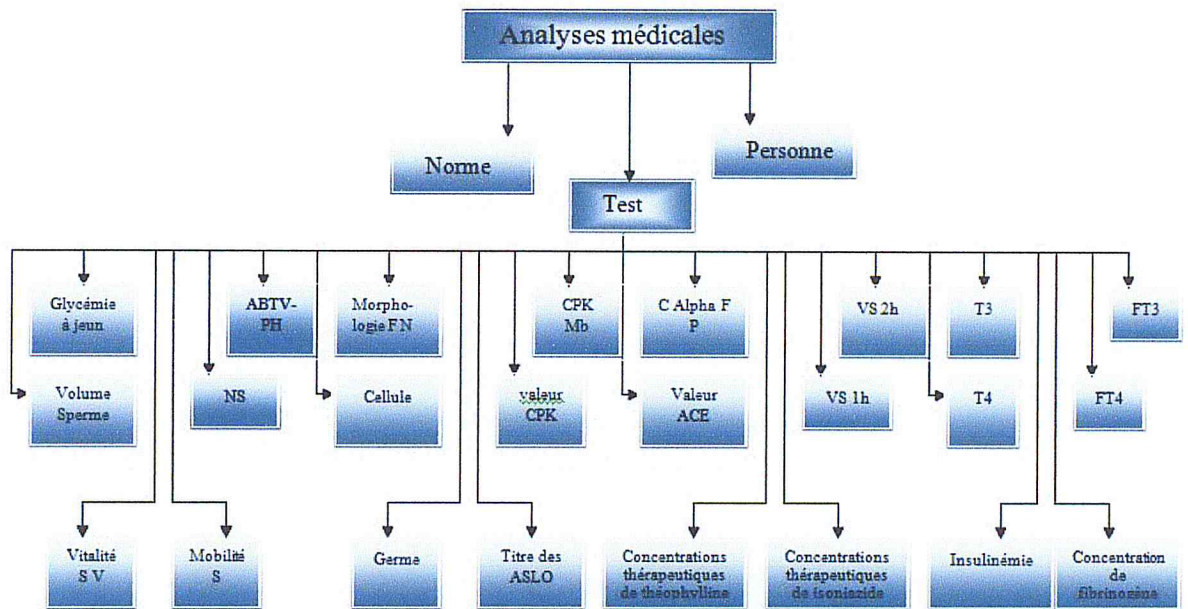


Figure III.22. Architecture de source XML.

- Une source données RuleML : c'est un document RuleML portant le nom « Examen-composer-test », voici un fragment de ce document :

```
<ruleml:rulebase>
  <Atom>
    <Rel>composer_de</Rel>
    <Var>Test</Var>
    <Var>ECB</Var>
  </Atom>
  <Atom>
    <Rel>composer_de</Rel>
    <Var>ECB</Var>
    <Var>Cellule</Var>
  </Atom>
  .....
</ruleml:rulebase>
```

Figure III .23 : Fragment de la source RuleML

Chaque source de données est associée a un schéma local ;

- **schéma local** : c'est le schéma d'une source de métadonnées, il représente les structures dans laquelle les métadonnées sont stockées dans cette source. Ces schémas sont représentés par des documents XML.

```
<?xml version="1.0" encoding="iso-8859-1"?>
Analyses_medicales xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="onto.xsd">

<Glycemie_a_jeun>
  <Norme>
    <Vmin>0.2</Vmin>
    <Vmax>0.6</Vmax>
    <Sexe>fh</Sexe>
    <Age_min>0</Age_min>
    <Age_max>13</Age_max>
  </Norme>
  <Norme>
    <Vmin>0.6</Vmin>
    <Vmax>1.10</Vmax>
    <Sexe>fh</Sexe>
    <Age_min>14</Age_min>
    <Age_max>100</Age_max>
  </Norme>
  <UM>g/l</UM>
  <Designation>Glycemie_a_jeun</Designation>

  </Glycemie_a_jeun>
```

**Figure III .24** : Un fragment de Schéma local de la base XML

```
<? xml version='1.0' encoding='iso-8859-1'?>
<composer_de>
<ECB>
<composer_de>
<Cellule></Cellule>
<Germe></Germe></composer_de></ECB>
<VS>
<composer_de>
<VS_1h></VS_1h>
<VS_2h></VS_2h></composer_de></VS>
<CPK>
<composer_de>
<Valeur_CPK></Valeur_CPK>
<CPK_MB></CPK_MB></composer_de></CPK>
<Insuline>
<composer_de>
<Insulinemie></Insulinemie></composer_de></Insuline>
<Spermogramme>
<composer_de>
<Volume_Sperme></Volume_Sperme>
<Vitalite_SV></Vitalite_SV>
<Mobilite_S></Mobilite_S></composer_de></Spermogramme>
.....
```

Figure III .25 : Un fragment Schéma local de la source RuleML



```
<Examen>
<Bacterio_Viro_Parasitologie>
<ECB></ECB>
<ASLO></ASLO></Bacterio_Viro_Parasitologie>
<Biologie>
<Glycemie></Glycemie>
<CPK></CPK></Biologie>
<Dosage_des_medicaments>
<theophylline></theophylline>
<isoniazide></isoniazide></Dosage_des_medicaments>
<Marqueurs_tumoraux>
<ACE></ACE>
<Alpha_FP></Alpha_FP></Marqueurs_tumoraux>
<Anatomo_pathologies>
<Spermogramme></Spermogramme></Anatomo_pathologies>
<Hematologie>
```

Figure III .26 : Un fragment Schéma local de la source RDF

### VI.1.2 Couche Adaptateur

L'adaptateur cache l'hétérogénéité au médiateur. Il est associé à une seule source et joue le rôle d'intermédiaire entre cette source et le médiateur.

C'est un « Traducteur » qui fait :

- La Traduction de la source de langage natif propre à la source en langage pivot.
- L'extraction des résultats de la sous requête envoyé par le médiateur.
- L'Envoie le résultat au médiateur.

- **Adaptateur RDF** : cet adaptateur est chargé de traduire la source **RDF** en une source **XML** et d'exécuter la requête sur la source traduite et de fournir le résultat au médiateur.
- **Adaptateur RuleML** : cet adaptateur est chargé de traduire la source **RuleML** en une source **XML** et d'exécuter la requête sur la source traduite et de fournir le résultat au médiateur.
- **Adaptateur XML** : est chargé d'interroger la source **XML** pour extraire une réponse.

### **VI.1.3 Couche Médiateur**

Il est décomposé en modules reliées entre eux. Pour bien répondre aux besoins d'un système d'intégration, un médiateur doit :

1. Accepter les requêtes des utilisateurs.
2. Localiser les sources de données pertinentes.
3. Décomposer la requête en sous requête.
4. Envoyer chaque sous requête au adaptateur correspondant.
5. Recomposer les résultats des adaptateurs et envoie le résultat à l'utilisateur.

Afin de bien réaliser ses taches, le médiateur possède les composants suivants, Qu'on va les détaillés par la suite :

1. Module création du schéma global.
2. Module de traitement de requêtes.
3. Module de recomposition des résultats.

#### ➤ **Schéma global :**

La présence d'un schéma global est nécessaire puisqu'il fournit un vocabulaire unique servant à exprimer les requêtes des utilisateurs. Ce schéma unifie les schémas hétérogènes des sources à intégrer en se basant sur une description homogène, uniforme et abstraite du contenu des sources par des vues.

Nous allons construire notre schéma global par l'approche GAV (Global as View), c'est-à-dire que le schéma global est considéré comme étant une vue sur les schémas des sources.

Notre schéma global est un fichier XML qui contient des vues sur chaque source, chaque vue est un ensemble de concept qui définit la source correspondante a cette vue.

```
<?xml version="1.0"?>
<shema_globale>
<vue_xml>
<person>
</person>
<test>
<g_a_j>
</g_a_j>
<vol_spr>
</vol_spr>
<vit_sv>
</vit_sv>
.....
<vue_xml>

<vue_rdf>
<exam>
<bvp>
<ecb>
</ecb>
<aslo>
</aslo>
</bvp>
<bio>
<glyc>
</glyc>
.....
</bio>
.....
</vue_rdf>
<vue_ruleml>
<ecb></ecb>
<aslo></aslo>
<glyc></glyc>
<ison></ison>
.....
</vue_ruleml>
```

Figure III .27 : Un fragment Schéma global

#### **VI.1.4 Couche utilisateur**

C'est une simple Interface de communication permet à un utilisateur de communiquer avec le système, Il envoie des requêtes au médiateur et reçoit des réponses, cette interface contient des champs de sélection qui aide l'utilisateur à sélectionner les éléments constituant la requête.

#### **VI.2 Description des modules**

##### **a. Le module Création de schéma global**

Afin de créer le schéma Global nous recherchons, pour chaque concept décrivant une relation ses équivalents dans les différentes sources de métadonnées a travers une table de correspondance.

**Correspondances entre schémas Global / Local (Global as View) :** L'approche Global as View est celle utilisée pour décrire les correspondances entre schémas (schéma global /schémas des sources). C'est une approche ascendante depuis les sources vers le médiateur.

Exemple

Fraction d'un Document décrivant la base de données

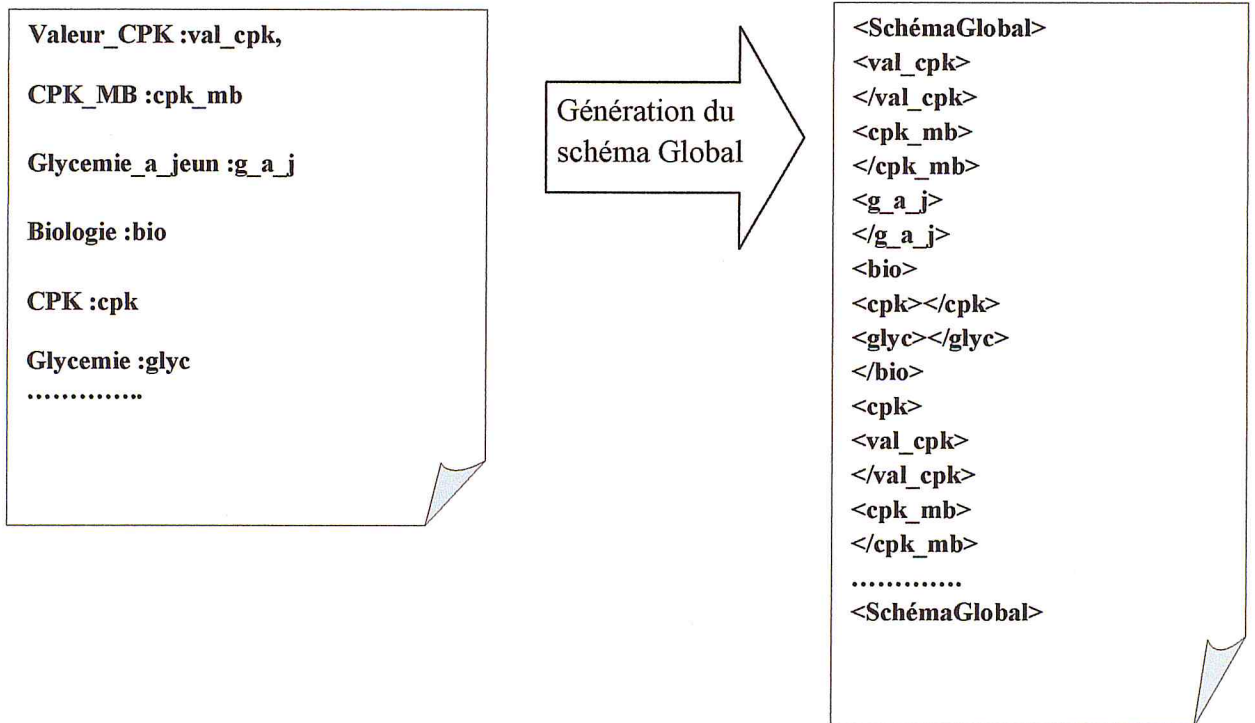
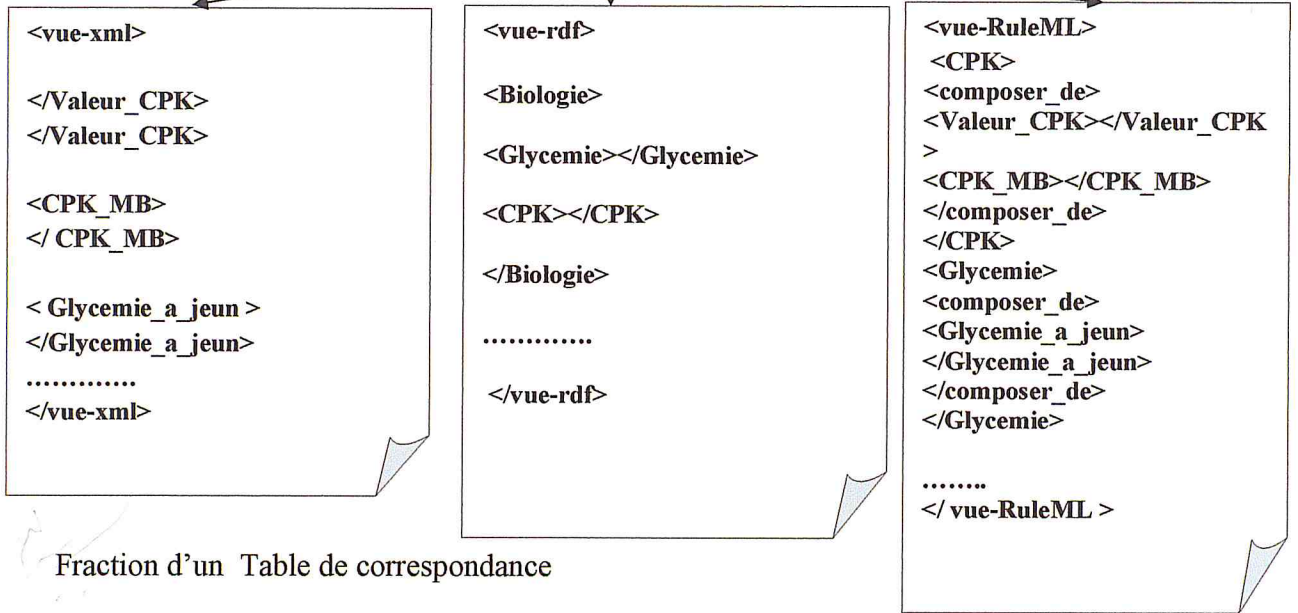


Figure VI .28 : Génération du schéma Global

❖ **Algorithme de Création du schéma Global**

**Algorithme : Création schéma global**

**Entrée :** Schémas locaux, Méta données *table de correspondance*

**Sortie :** Schéma global

**Début**

Connecter aux sources de données

**Si** connexions établit **alors**

Consulter les schémas locaux ✓

Extraire l'ensemble de concepts locaux  $E_1$

**Pour chaque** concept local  $CL_i$  de l'ensemble  $E_1$  **faire**

Chercher son synonyme  $SC_i$  dans la table de correspondances

**Si**  $SC_i$  existe **alors** concept global  $CG_i$  de  $CL_i$  sera  $SC_i$

**Sinon**  $CG_i$  sera lui-même

Grouper les  $CL_i$  qui ont le même  $CG_i$

**Pour chaque** attribut local  $AL_i$  de l'ensemble  $E_2$  **faire**

Chercher son synonyme  $SA_i$  dans la table de correspondance

**Si**  $SA_i$  existe **alors** attribut global  $AG_i$  de  $AL_i$  sera  $SA_i$

**Sinon**  $AG_i$  sera lui-même

Grouper les  $AL_i$  qui ont le même  $AG_i$

**Sinon** rien faire

**Fin**

**b. Le module Traitement de requête**

Le processus traitement de requête passe par trois phases principales :

**b.1 Décomposition de requête et Localisation des sources**

Ce module est chargé de décomposer une requête globale (de l'utilisateur) en sous requêtes définies sur une relation globale.

Formellement, considérons une requête  $Q$  dont la syntaxe est la suivante :

**Exemple**

```
for $c in doc ("nom-fichier-xml")/racine
  for $a in $c/cpt-globale
return
  { $a
  }
```

On à Domain-examen se compose de plusieurs examens médicaux et chaque examen médical se compose de plusieurs tests.

Donc on s'intéresse à extraire le résultat de chaque test d'un examen médical dans tel domaine pour une personne donnée.

Tel que :

**Domain-examen** : se trouve dans la source de métadonnées RDF.

**Examen-médical** : se trouve dans la source de métadonnées RuleML.

**Test** : se trouve dans la source XML.

**Personne** : se trouve dans la source XML.

**Exemple** : Afin de faciliter la compréhension de notre algorithme, considérons l'exemple suivant. Soit la requête utilisateur suivante :

```
for $c in doc ("schema-globale.xml")/Analyses-medicals
  for $a in $c/bio
return
  { $a
  }
```

Cette dernière peut être décomposée en trois sous requêtes définie sur le schéma global, chacune va être destinée à une source.

Sous Requête 1: 

```
for $c in doc ("source-RDF.xml")/Analyses-medicals
  for $a1 in $c/bio
  return
  {$a1}
```

Sous Requête 2: `for $c in doc ("source-RuleML.xml")/ Analyses-medicals`  
`for $a2 in $c/glyc`  
`return`  
`{$a2}`

Sous Requête 3: `for $c in doc ("source-XML.xml")/ Analyses-medicals`  
`for $a3 in $c/g-a-j`  
`return {$a3}`

## **b.2 Localiser les sources pertinentes :**

Ce module a pour rôle d'identifier les relations locales pertinentes ainsi que leurs sources.

A partir de la requête on peut localiser directement la source pertinente

### **Exemple :**

```
for $c in doc ("nom de la source")/Analyses-medicals
  for $a in $c/cpt-globale
  return
    { $a
    }
```

## **c. Le module Recomposition des résultats**

Ce module construit la réponse d'une requête globale en utilisant les résultats des requêtes locales envoyées par les adaptateurs. Nous présentons l'algorithme de reconstruction de la réponse d'une sous requête  $Q_i$ .



**Algorithme : Recomposition des sous résultats**

**Entrée :** Un ensemble de sous réponse

**Sortie :** réponse globale

**Début**

Pour chaque sous réponse SR **Faire**  
Inserer dans la réponse Global RG, SR

**Fait**

**Fin**

## **VII. Conclusion**

Dans ce chapitre nous avons présenté une conception détaillée de notre système d'intégration basé sur les métadonnées en utilisant le langage UML, cette étude conceptuelle nous a permis de mettre en évidence les étapes nécessaires pour la création d'un système d'intégration des sources de métadonnées.

La prochaine étape consiste donc en la concrétisation de ce que nous avons proposé, en d'autres termes, la réalisation d'un système d'intégration basé sur les métadonnées et les outils que nous utilisons pour l'implémenter.

**CHAPITRE IV**  
**IMPLEMENTATION**  
**ET TEST**

## I. Introduction

Après l'expression des besoins et la modélisation du système à développer, nous allons, dans cette partie, présenter son implémentation et sa mise en œuvre. Nous présenterons chaque source de métadonnées (XML, RDF, RuleML) avec l'outil utilisé pour la traiter, ensuite l'environnement de développement utilisé dans la réalisation du système, puis les différentes interfaces utilisateur, administrateur et les fonctionnalités de chacune.

## II. Réalisation

### II.1. Implémentation des sources

#### II.1.1. Classes et hiérarchies des classes

Dans ce qui suit nous décrivons les classes de nos sources de métadonnées :

- **Analyses médicales** : Classe abstraite qui regroupe les examens, les tests et les normes.
- **Examen** : Classe abstraite regroupant plusieurs familles d'examens (Bactério Viro Parasitologie, Biologie, Dosage des médicaments, Marqueurs tumoraux et Anatomopathologies) qu'on les appelle **Domaine d'examen**. Chaque famille est composée de plusieurs examens qui sont représentés par des classes concrètes (ASLO, ECB, Glycémie, CPK, Théophylline, Isoniazide, ACE, Alpha F P, Spermogramme, Fibrinogène, VS, T3 T4 FT3 FT4...).
- **Test** : Chaque examen possède plusieurs tests (exemple : L'examen « CPK » qui appartient à la famille « Biologie » contient les deux tests : « CPK Mb des CPK totales » et « Valeur normale CPK »).
- **Norme** : Classe qui représente un ensemble de caractéristiques décrivant un test. Tout ce qui entre dans une norme est considéré comme « normal », alors que ce qui en sort est « anormal » (Pour l'exemple précédant de « CPK » chaque test a ses propres normes tel

que : le CPK Mb des CPK totales doit être inférieur à 30 % et la valeur normale CPK doit être entre 10 et 100 %).

## **II.1.2. Source de métadonnées RDF**

### **II.1.2.1. Altova SemanticWorks 2011**

Est un éditeur RDF / OWL pour le Web sémantique. Graphiquement il peut dessiner et éditer des documents d'instance RDF, RDF Schéma (RDFS) vocabulaires, ontologies OWL, puis la sortie du format de votre choix. SemanticWorks rationalise le travail avec la syntaxe complète et la vérification sémantique. En outre, les aides de l'entrée contextuelle vous présenter une liste de choix autorisés sur la base des RDF ou OWL dialecte que vous utilisez, si vous êtes sûr de créer des documents valides à chaque fois. Dialectes pris en charge incluent RDF, RDF Schéma, OWL Lite, OWL DL et OWL Full. Onglets pour les classes, propriétés, instances, etc. Une fenêtre Détails pratiques d'assistance d'entrée vous permet de créer de nouvelles instances de RDFS ou OWL classes et leur attribuer leurs propriétés rapidement et facilement. Comme vous concevez, vous pouvez passer du graphique RDF ou OWL en vue de l'affichage de texte pour voir comment votre document est en cours de création dans les deux RDF / XML ou en format N-triples, et vous pouvez exporter votre fichier de RDF / XML N-triples ou vice versa, à tout moment. Qui plus est, car le code RDF / XML ou N-triples est généré automatiquement en fonction de votre design, vous pouvez expérimenter avec les concepts du Web sémantique et d'apprendre sans avoir à écrire du code compliqué. [ALT, 2010].

### **II.1.2.2. La représentation de la source de métadonnées RDF sur l'éditeur Altova SemanticWorks 2011**

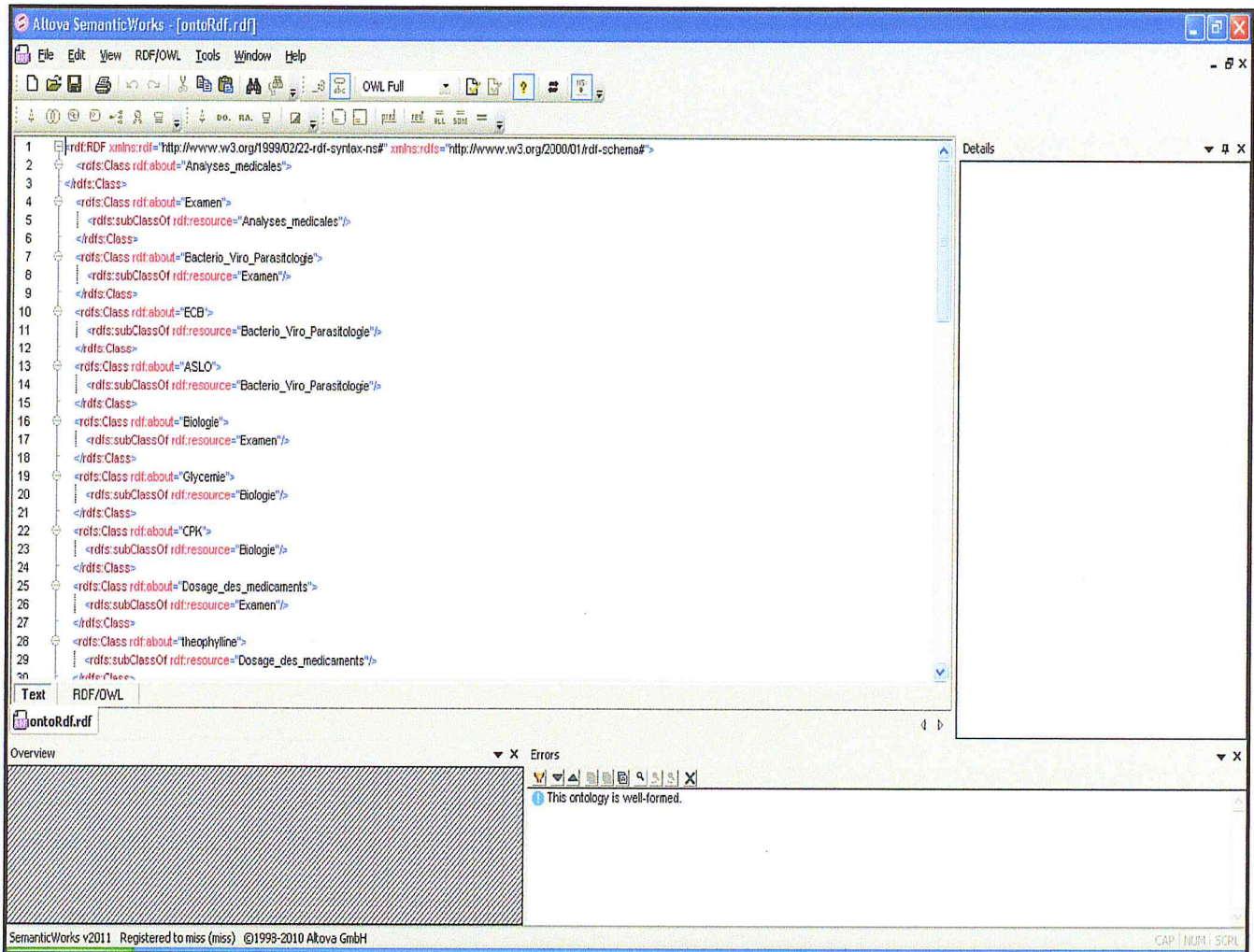


Figure IV.1 : Le contenu de notre Source de métadonnées RDF sur Altova SemanticWorks 2011.

### II.1.3. Source de métadonnées XML

#### II.1.3.1. Stylus Studio 2011 XML Enterprise

Offre un ensemble complet d'outils XML et des fonctionnalités pour travailler avec XML, XQuery, les services Web XML édition, et de nombreuses autres technologies XML. Il inclut un XML Validateur, il offre aussi une gamme complète de soutien le développement XSLT, y compris le débogage XSLT, cartographie XSLT, le profilage XSLT [STY, 2011].

### II.1.3.2. La représentation de la source de métadonnées XML sur l'éditeur Stylus Studio 2011

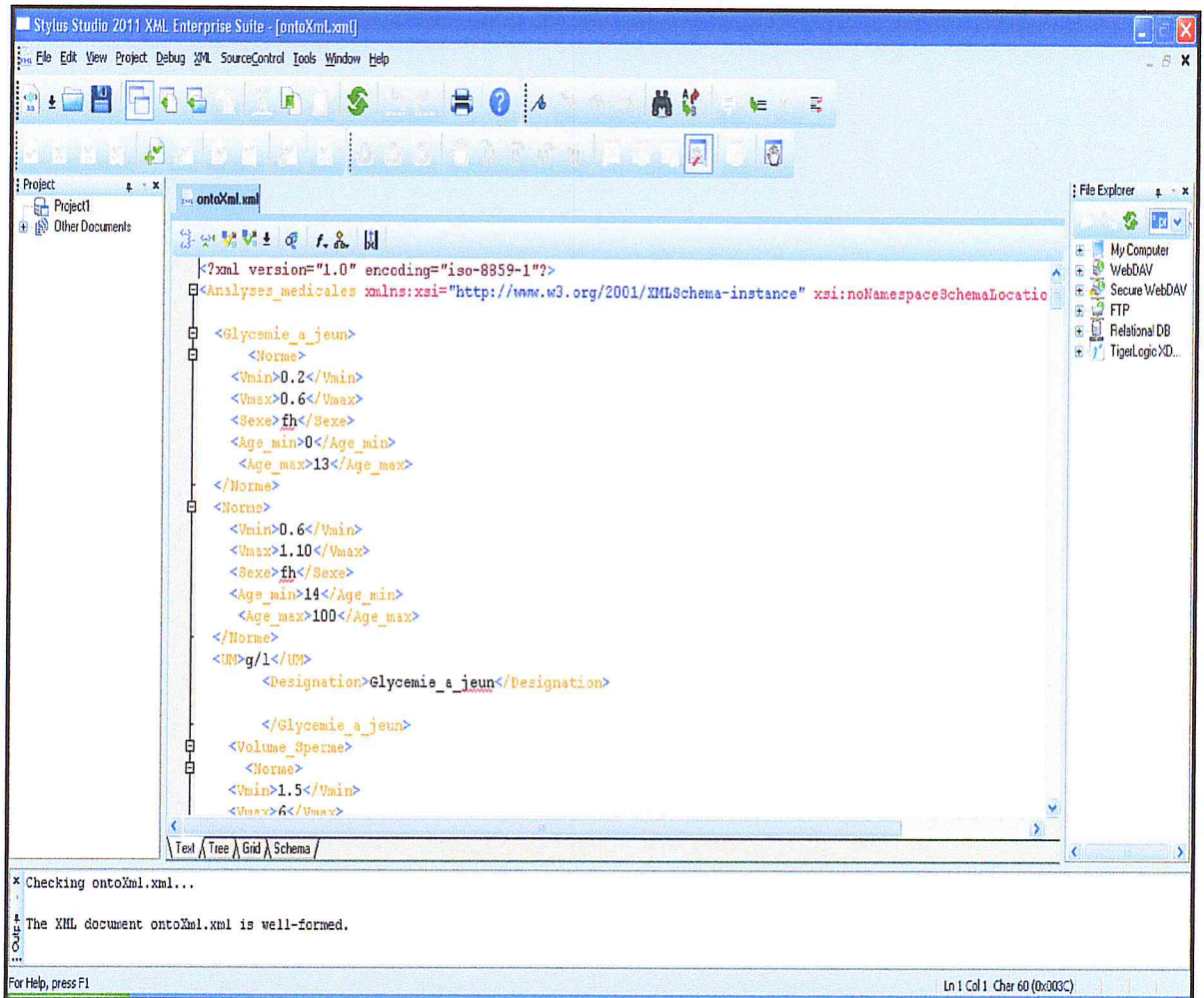


Figure IV.2 : Le contenu de notre Source de métadonnées XML sur Stylus Studio 2011.

### II.1.4. Source de métadonnées RuleML

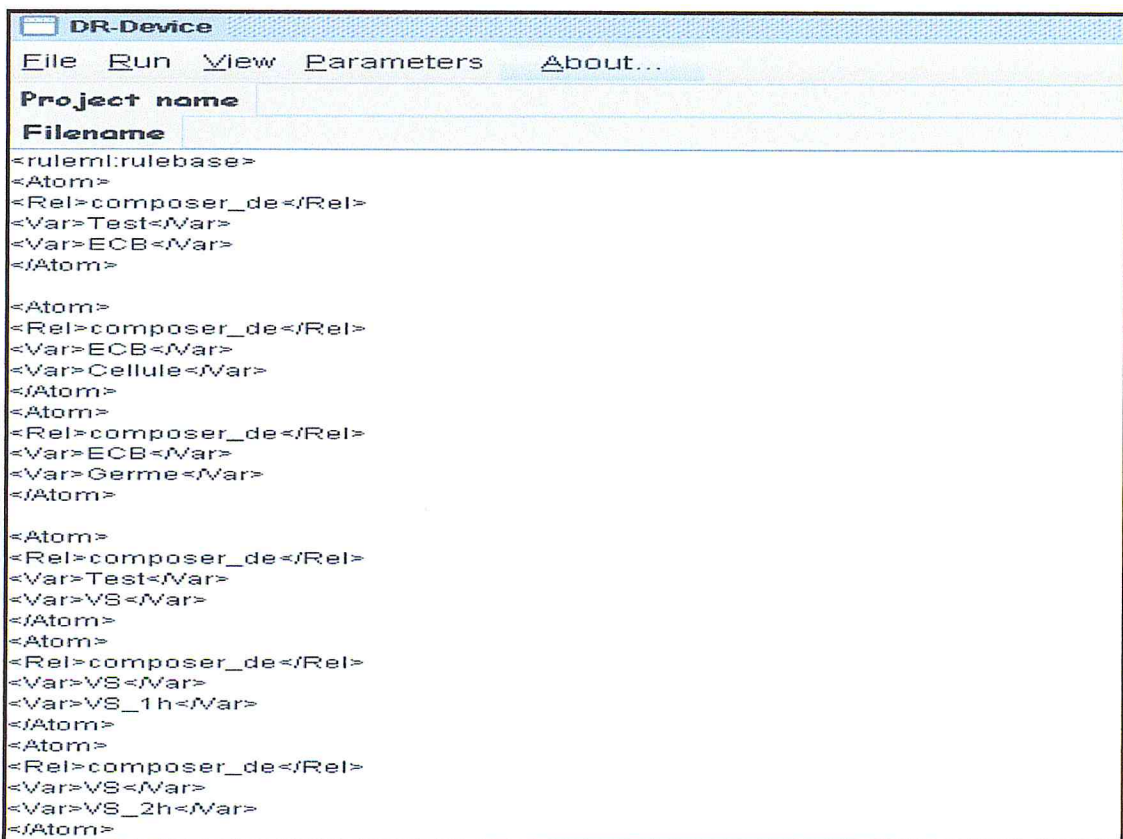
Notre source **RuleML** définit la relation de composition entre les examens et leurs tests.

Par exemple : L'examen « T3-T4-FT3-FT4 » est composé de 4 test qui sont : t3, t4, ft3, et ft4.

### II.1.4.1. DR-Device

Un environnement de développement visuel intégré pour le développement et l'utilisation de bases annulable règle logique sur le dessus d'ontologies RDF. un éditeur de règles visuelle RuleML-conforme qui contraint le vocabulaire autorisé par l'analyse des ontologies RDF entrée et un système de raisonnement révisable que les processus de données RDF et des ontologies RDF Schéma. [LIN]

### II.1.4.2. La représentation de la source de métadonnées RuleML sur l'éditeur DR-Device



```
<code>DR-Device
File Run View Parameters About...
Project name
Filename
<ruleml:rulebase>
<Atom>
<Rel>composer_de</Rel>
<Var>Test</Var>
<Var>ECB</Var>
</Atom>

<Atom>
<Rel>composer_de</Rel>
<Var>ECB</Var>
<Var>Cellule</Var>
</Atom>
<Atom>
<Rel>composer_de</Rel>
<Var>ECB</Var>
<Var>Germe</Var>
</Atom>

<Atom>
<Rel>composer_de</Rel>
<Var>Test</Var>
<Var>VS</Var>
</Atom>
<Atom>
<Rel>composer_de</Rel>
<Var>VS</Var>
<Var>VS_1h</Var>
</Atom>
<Atom>
<Rel>composer_de</Rel>
<Var>VS</Var>
<Var>VS_2h</Var>
</Atom>
</code>
```

Figure IV.3 : Le contenu de notre Source de métadonnées RuleML sur DR-Device

## II.2. Mécanismes de manipulation des documents XML

### II.2.1. Modèle objet de document: DOM

Le DOM (Document Object Model) est une interface de programmation (API) pour des documents XML ou HTML. Cette interface est indépendante des plates-formes et des langages de programmation. Elle doit permettre aux scripts et aux programmes d'accéder et mettre à jour dynamiquement le contenu, la structure et la présentation d'un document XML. Les documents sont modélisés en mettant en évidence la structure du document, son comportement et les objets qui les composent. Les nœuds d'un document ne représentent pas une structure de données mais des objets, ayant une fonction et une identité [BAK, 2006].

Le DOM identifie:

- les interfaces et objets utilisés pour représenter et manipuler un document,
- la sémantique de ces interfaces et objets (comportement et attribut),
- les relations et les collaborations entre les interfaces et les objets.

Le DOM fournit:

- un ensemble d'objets standards pour représenter un document XML
- un modèle de référence présentant les liens entre ces objets.
- une interface standard pour y accéder et les manipuler.

### II.2.2. Interrogation de documents XML en utilisant XQuery

XQuery est un langage déclaratif analogue au langage SQL mais SQL manipule des tables (ensemble de n-uplets), XQuery manipule des séquences d'arbres.

XQuery est un langage puissant mais complexe à mettre en oeuvre à cause de la complexité des structures manipulées (les arbres XML) et la semi-structuration de ces données (problèmes de typage) [BAK, 2006].

Une requête XQuery est une composition d'expressions. Chaque expression a une valeur ou retourne une erreur. Les expressions n'ont pas d'effets de bord (par exemple, pas de mise à jour).

La forme de requête est FLWOR (For-Let-Where-Order-Return) dans des forêts

**for** \$<var1> in <forest1> [, \$<var2> in <forest2> ]...// itération



**let** \$<varn>:= <subtree> // assignation  
**where** <condition> // élagage  
**order-by** \$var1 // ordonnancement  
**return** <result> // construction

- F: Collection d'arbres utilisés équivalent du FROM de SQL;
- L: Mémorisation d'arbres et affectation de variables locales;
- W: Condition (élagage) équivalent du WHERE de SQL;
- O: Ordonnancement équivalent de ORDER-BY de SQL;
- R: Sous-arbres sélectionnés, Présentation des sous-arbres équivalent du SELECT de SQL avec une reconstruction [BAK, 2006]

XQuery est un langage fonctionnel, Une requête XQuery est une expression qui est évaluée dans un certain contexte. La valeur d'une expression est une séquence. Une séquence est une collection de 0 ou plusieurs items. Un item est une valeur atomique ou un nœud de l'arbre d'un document. [BAK, 2006]

## II.3. Le langage de programmation JAVA

### II.3.1. Java est un langage orienté objet

C'est un langage de programmation orienté objet, développé par Sun Microsystems. Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris). Java donne aussi la possibilité de développer des programmes pour téléphones portables et assistants personnels. Enfin, ce langage peut être utilisé sur internet pour des petites applications intégrées à la page web (applet) ou encore comme langage serveur (jsp).

Les avantages de Java sont nombreux. Le byte-code, tout d'abord, qui assure à Java une portabilité complète vers de très nombreux systèmes. L'importance des API de base qui offre tous les services de base, notamment pour la construction des interfaces graphiques. La 3<sup>ème</sup> force de Java, c'est son adaptabilité dans de nombreux domaines, autant pour le web que pour les systèmes embarqués. [JAV ,2011]

### II.3.2. Le SGBD MYSQL

Un serveur de bases de données stocke les données dans des tables séparées plutôt que de tout rassembler dans une seule table. Cela améliore la rapidité et la souplesse de l'ensemble. Les tables sont reliées par des relations définies, qui rendent possible la combinaison de données entre plusieurs tables durant une requête. Le SQL dans "MySQL" signifie "Structured Query Language" : le langage standard pour les traitements de bases de données [MQL, 2011].

### II.4. Démonstration

L'accès à notre Système peut se faire de deux manières :



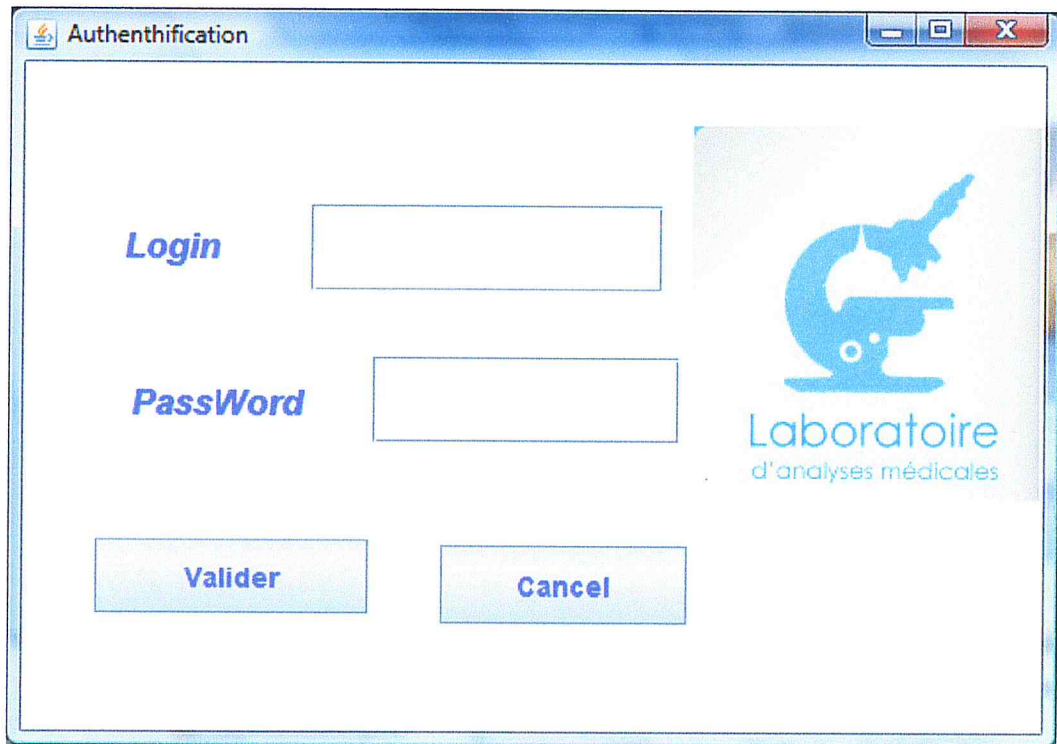
Figure IV.4 : page d'accueil de système.

✓ **Accès Utilisateur:**

Pour les utilisateurs qui ne disposent pas des droits d'accès administrateur, ils peuvent accéder à notre système de manière restreinte, par exemple ils ne peuvent pas voir la génération de la requête ils peuvent juste voir le résultat final.

✓ **Accès administrateur:**

L'administrateur s'identifie en saisissant, dans la zone d'identification (voir: **Figure IV.5**), un login et un mot de passe. Si ces informations existent dans la base de données de Système; le système affiche la page correspondante a l'administrateur



The image shows a screenshot of a software window titled "Authentification". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. Inside the window, there are two text input fields. The first field is labeled "Login" and the second is labeled "PassWord". Below these fields are two buttons: "Valider" (Validate) and "Cancel". To the right of the input fields, there is a logo for "Laboratoire d'analyses médicales" which includes a stylized blue microscope icon.

Figure IV.5 : page d'authentification.

### L'administrateur

- **Page d'accueil** : Dans cette page l'administrateur a le droit de :
  - Créer le schéma global.
  - Ouvrir une source de métadonnées.
  - Faire les mises à jour pour les métadonnées de chaque source.
  - Poser une requête.

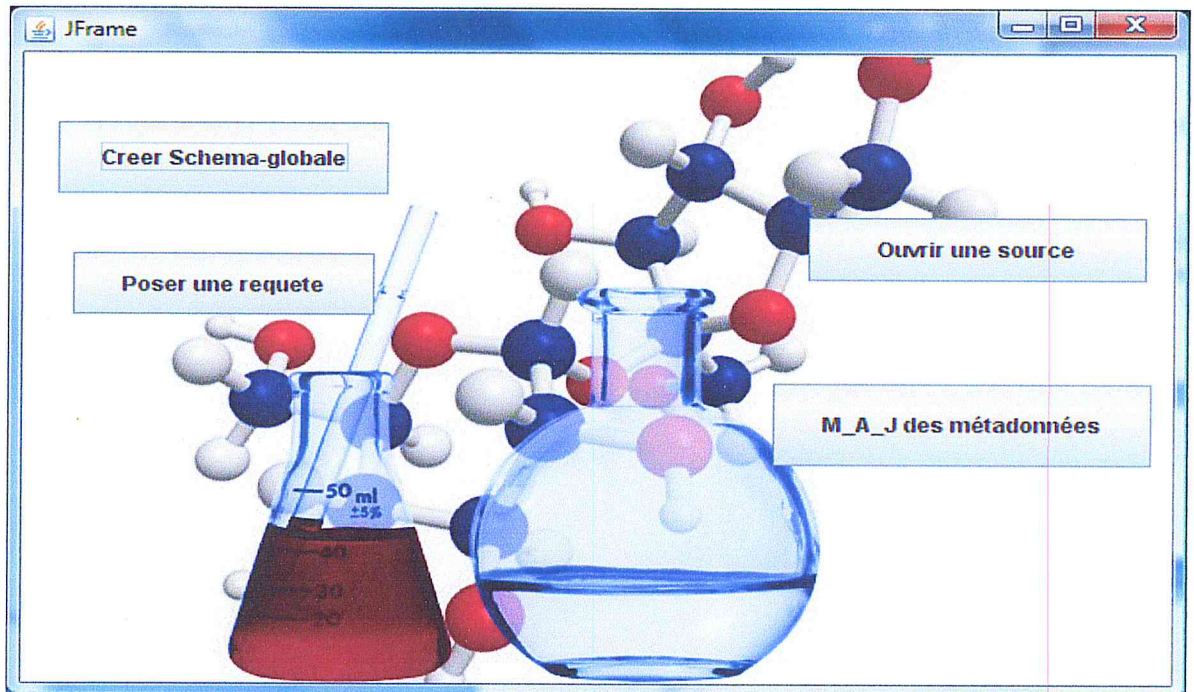


Figure IV.6 : page d'accueil de l'administrateur.

- **Créer schéma globale :**

L'administrateur choisi les sources pour créer un schéma globale sur ses derniers

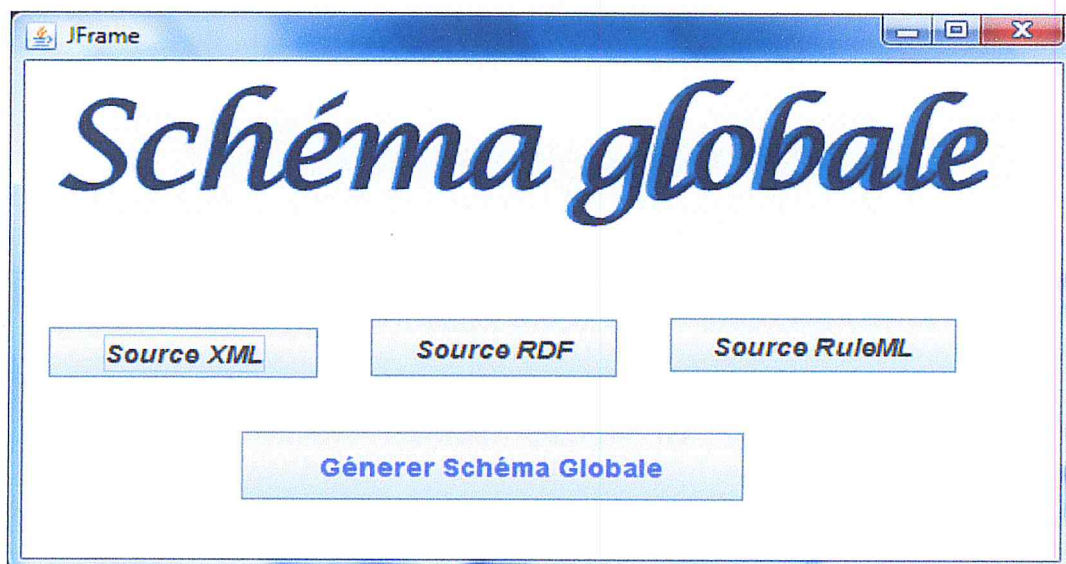


Figure IV.7 : Créer un schéma global.

- **schéma global :**

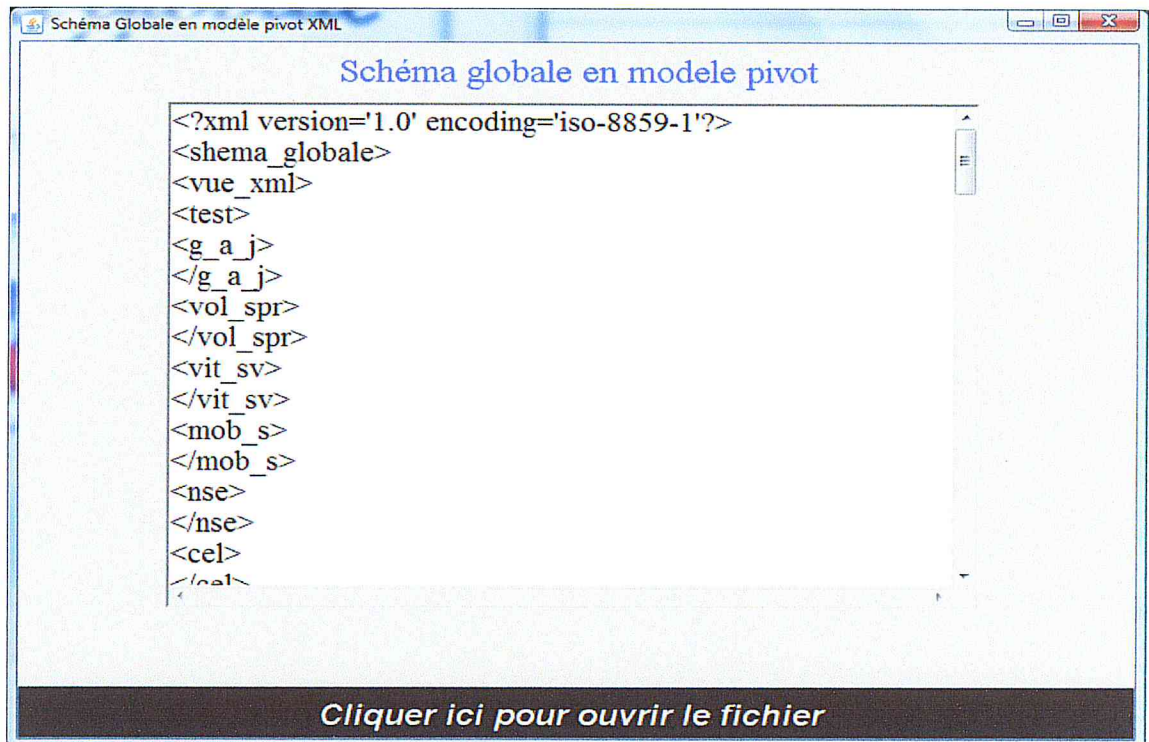


Figure IV.8: génération de schéma global.

- **Poser une requête :**

Avant de poser une requête l'administrateur doit sélectionner son ID, une fois l'ID est sélectionné, le système génère une requête XQuery qui fait l'extraction des coordonnées de la personne qui correspond a cet ID et affiche ces dernières (Voir : la Figure IV.9).

L'administrateur peut poser sa requête soit :

- Par Domaine.
- Par Examen.
- Soit par Test.

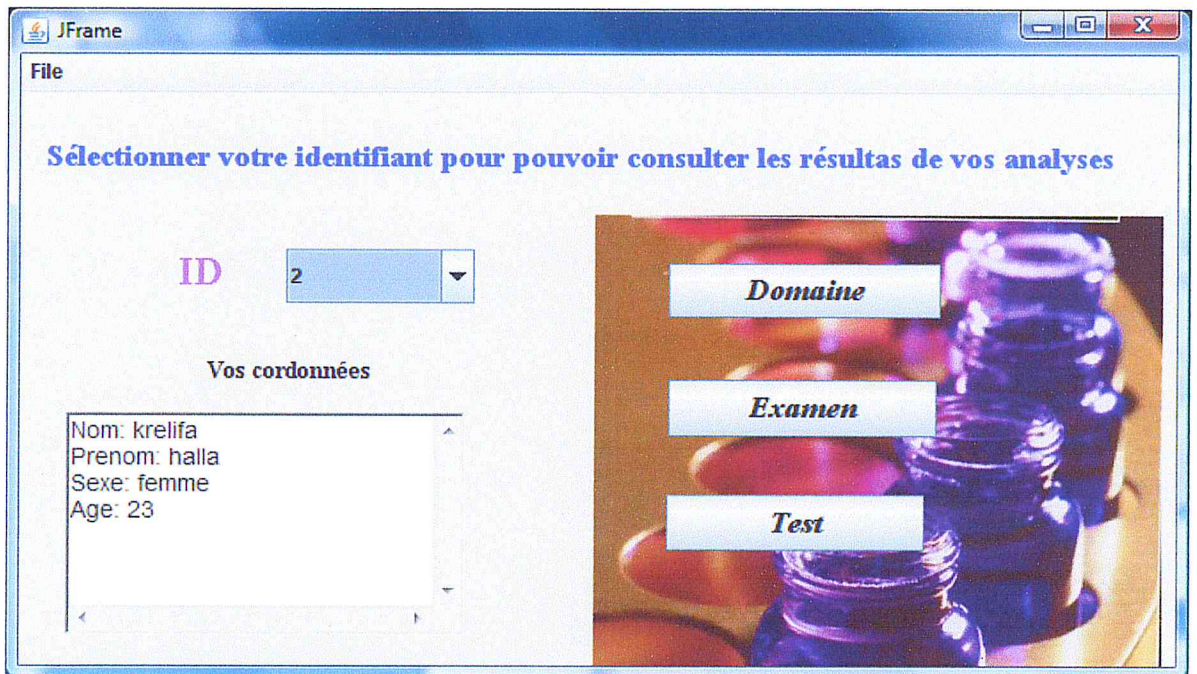


Figure IV.9: page pour poser une requête.

### Par domaine

L'administrateur a le choix de sélectionner certain domaine ou bien tous les domaines.

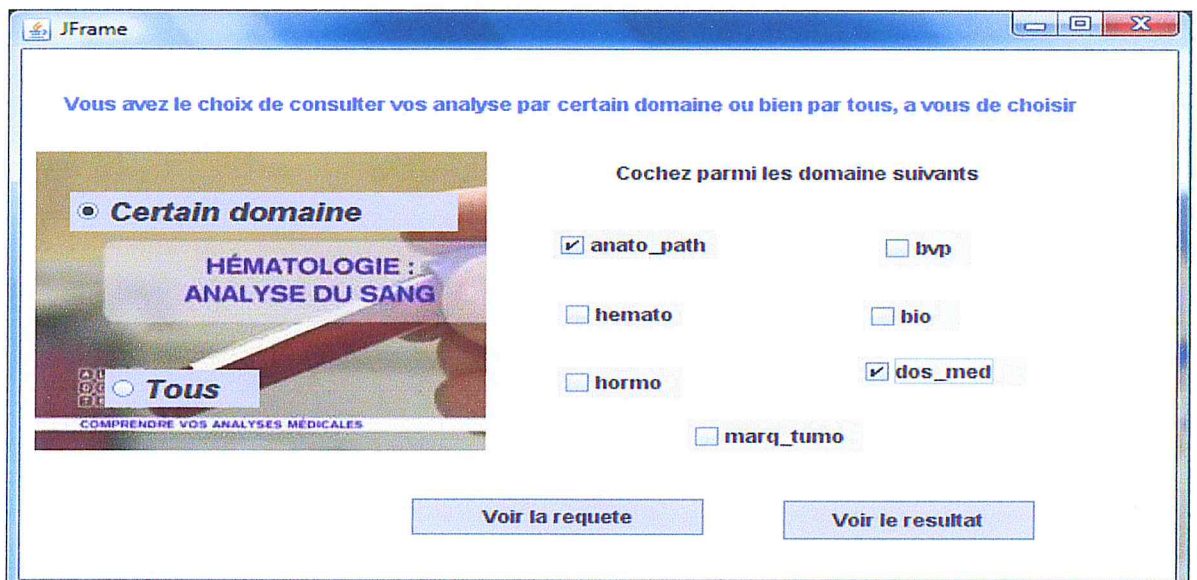


Figure IV.10: page pour poser une requête par domaine.

### Voir la requête

Une fois la sélection des domaines est faite, l'administrateur peut voir la requête générée sous format XML

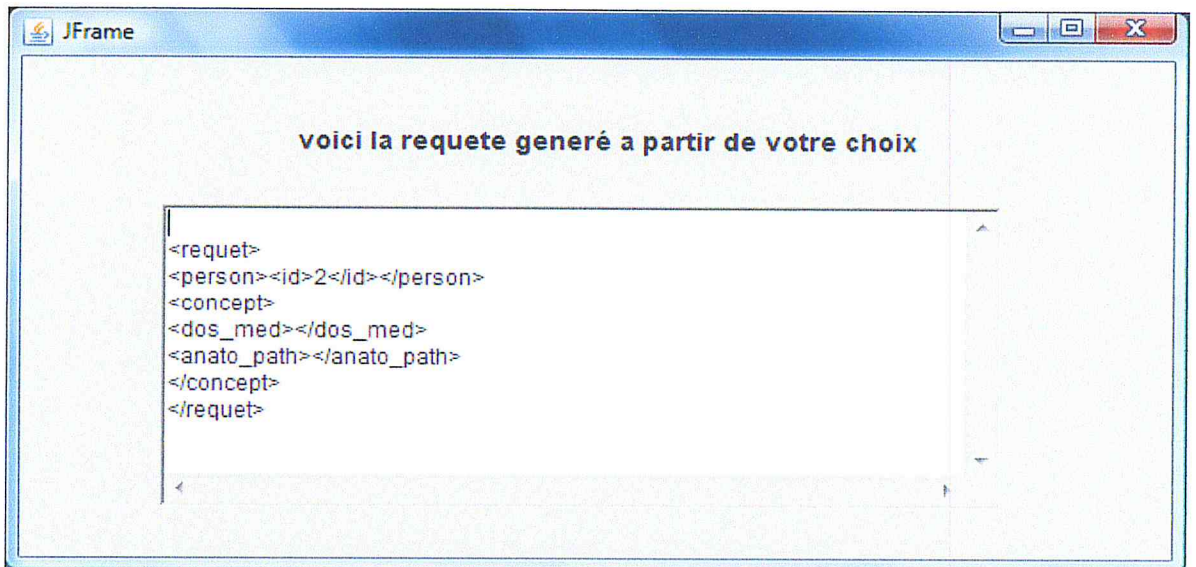


Figure IV.11: voir la requête générée.

### Traitement de la requête

Quand l'administrateur choisi un domaine donc il veut les valeurs de tous les tests de chaque examen de ce domaine choisi, sachant que :

- Les domaines se trouvent dans la source RDF.
- Les examens de chaque domaine se trouvent dans la source RuleML.
- Les tests de chaque examen se trouvent dans la source XML.

Donc une fois la requête est générée, elle doit être décomposée en trois sous requête :

- Une sous requête pour l'extraction de domaine.
- Une sous requête pour l'extraction des examens de domaine choisi.
- Une sous requête pour l'extraction des tests de l'examen de domaine choisi.

### Voir le résultat

Après la sélection des domaines l'administrateur peut voir le résultat de la requête générée par le système en suivant les domaines choisis.

### Exemple :

Dans ce cas l'administrateur choisit le domaine « **dos\_med** » :

1. À partir de la table de correspondance (qui contient pour tout concept global son concept local) on extrait le concept local de concept global « **dos\_med** » qui est « **Dosage\_des\_médicaments** » qui se trouve dans la source RDF.
2. a partir de la source RuleML on extrait toutes les examens de domaine « **Dosage\_des\_médicaments** » qui sont **théophylline** et **isoniazide**.
- 3.1. Pour l'examen **théophylline** on extrait son test «**Concentration\_therapeutiques\_de\_theophylline** » qui se trouvent dans la source XML.
- 3.2. Pour l'examen **isoniazide** on extrait son test «**Concentration\_therapeutiques\_de\_isoniazide** » qui se trouvent dans la source XML
4. Pour chaque test on fait l'extraction de la norme selon le sexe et l'âge de la personne qu'on a sélectionné son ID au début.
5. Enfin le Système fait la fusion des résultats obtenus et affiche les valeurs des tests avec leurs normes.
  - Si la valeur de test pour cette personne est dans la norme on affiche un message «**vous êtes dans la norme** »
  - Sinon le système affiche un message « **vous n'êtes dans la norme**»



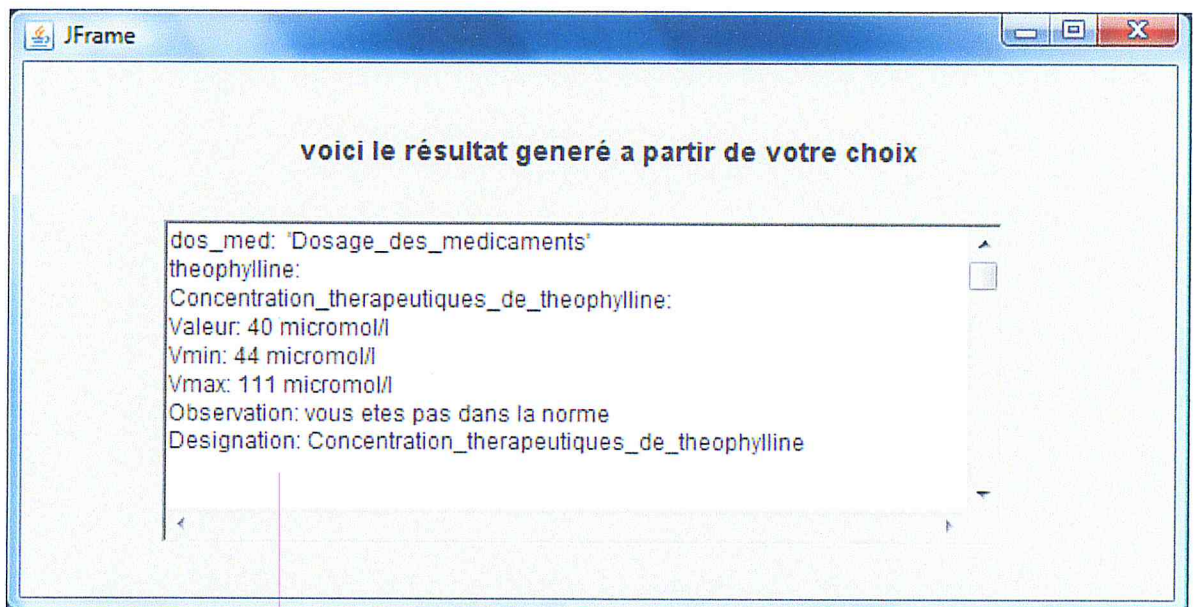


Figure IV.12: voir le résultat de la requête généré.

### Par Examen

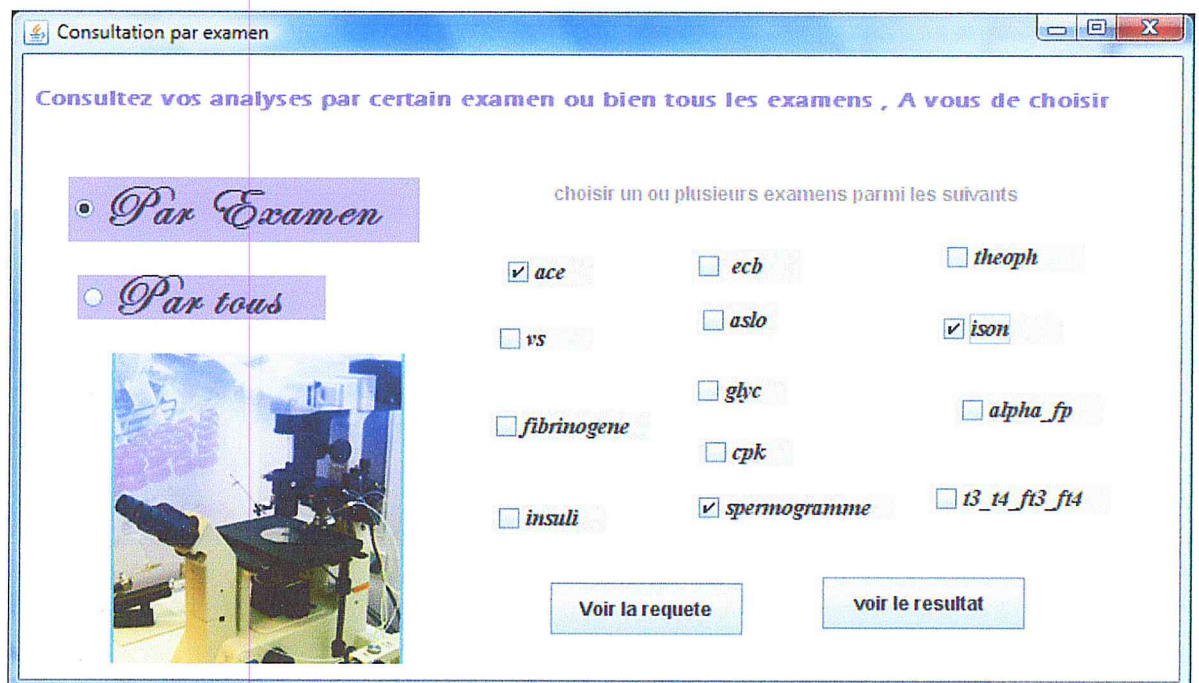


Figure IV.13: page pour poser une requête par Examen.

### Voir la requête

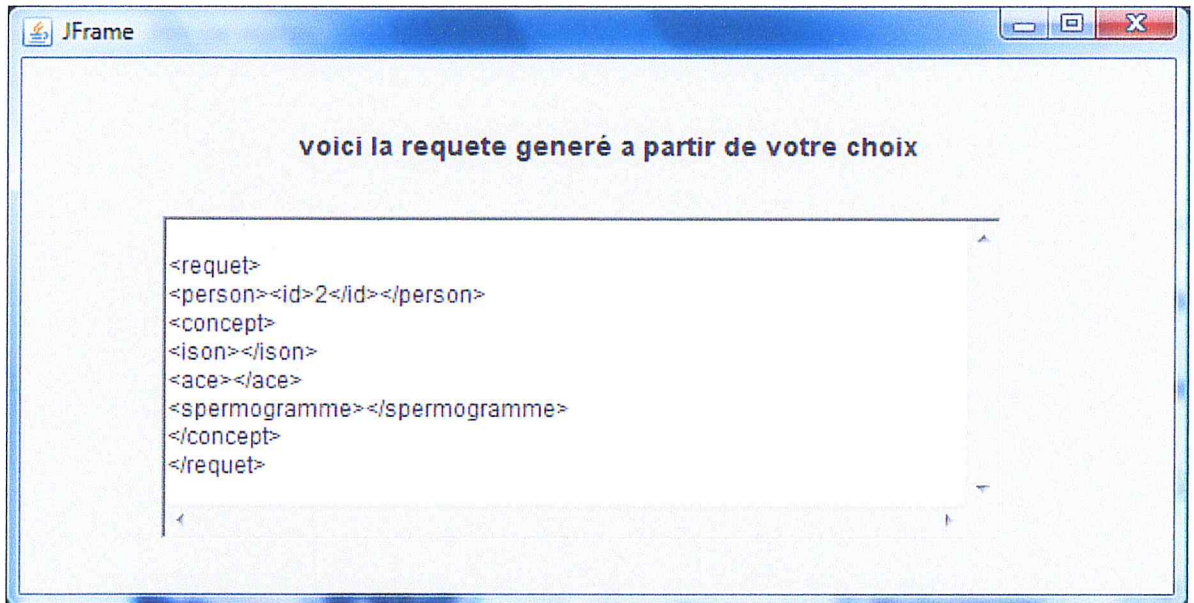


Figure IV.14: voir la requête générée.

### Voir le résultat

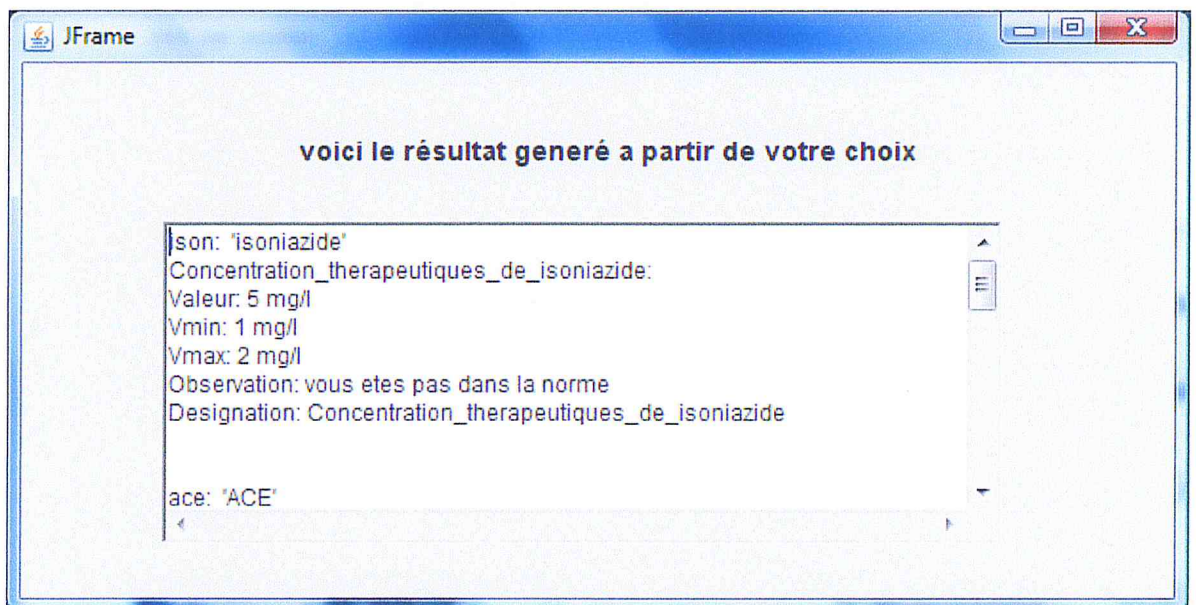


Figure IV.15: voir le résultat de la requête générée.

• *Par Test*

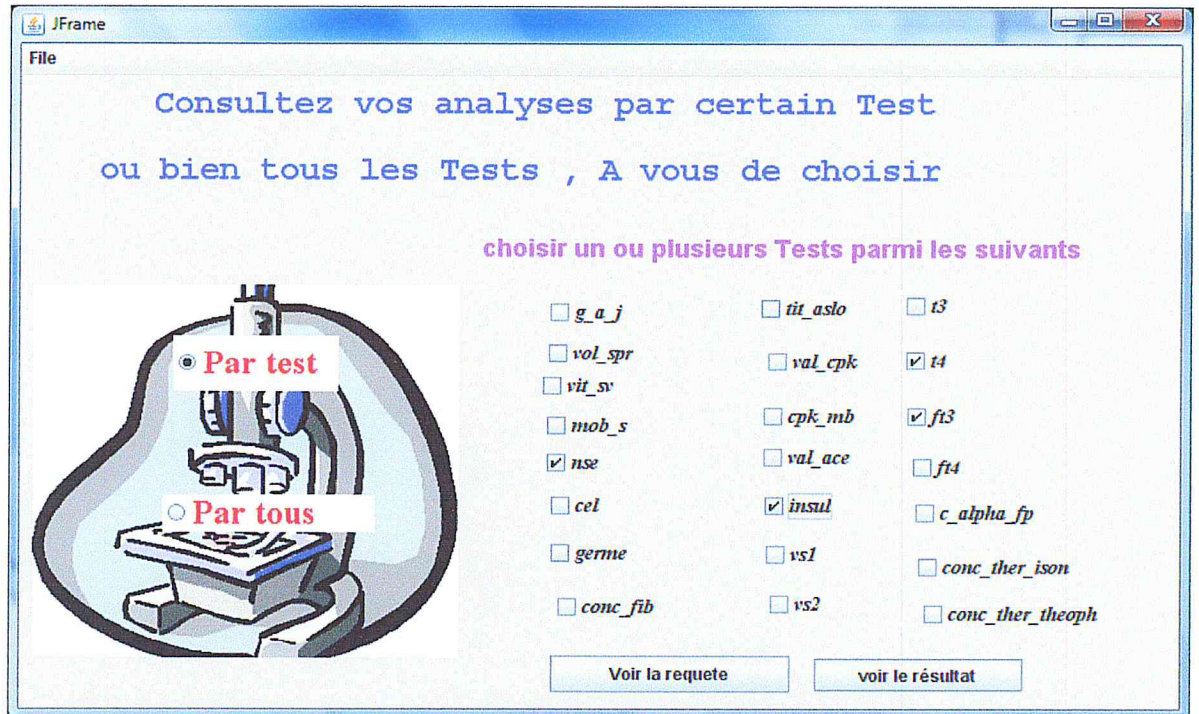


Figure IV.16: page pour poser une requête par Test.

**Voir la requête**

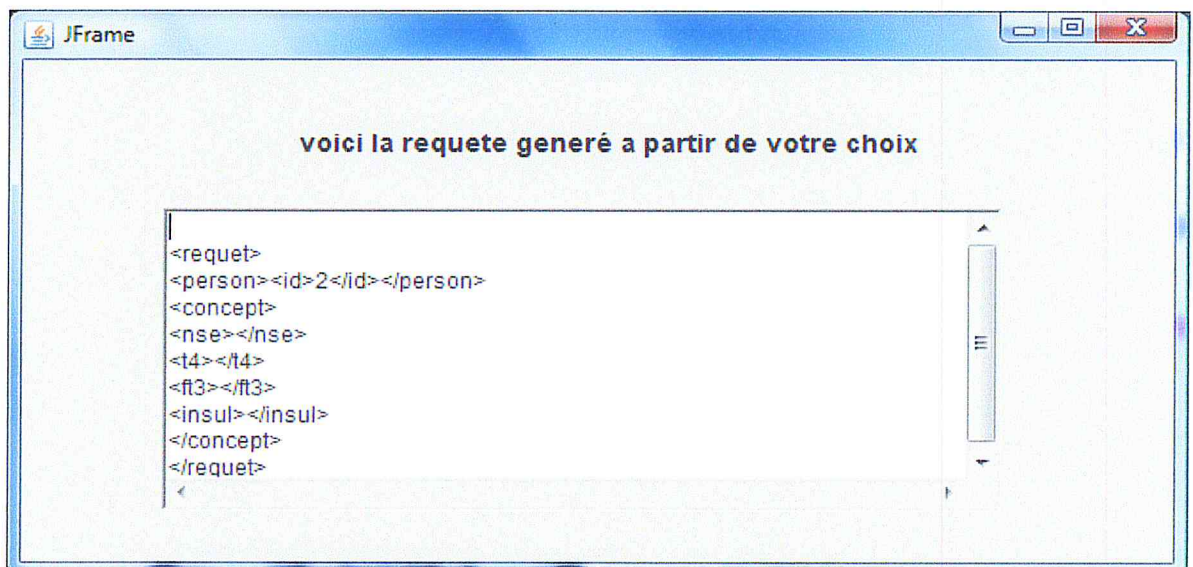


Figure IV.17: voir la requête générée.

### Voir le résultat

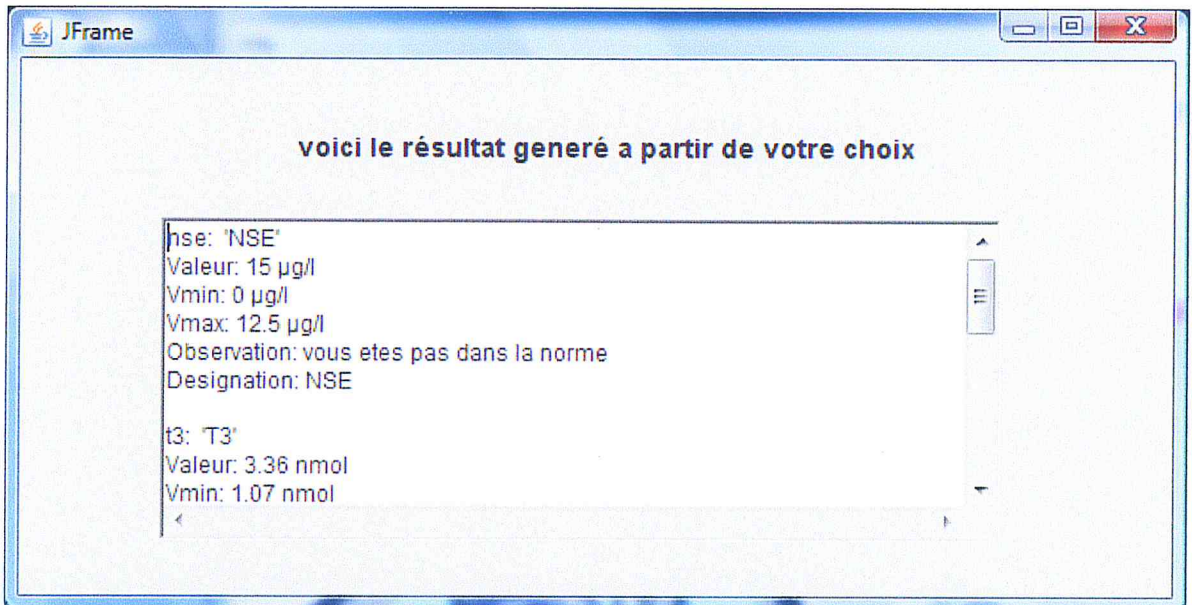


Figure IV.18: voir le résultat de la requête générée.

- **La mise à jour des métadonnées**

L'administrateur peut ajouter, supprimer, modifier une métadonnée dans chaque source.

**Ajouter une métadonnée :**

Il choisit la source dans laquelle il veut ajouter une métadonnée, ensuite saisit le nom de la métadonnée et clique sur le bouton ajouter.

**Modifier une métadonnée :**

Il choisit la source dans laquelle il veut modifier une métadonnée, ensuite saisit le nom de la métadonnée et clique sur le bouton modifier.

**Supprimer une métadonnée :**

Il choisit la source dans laquelle il veut supprimer une métadonnée, ensuite saisit le nom de la métadonnée et clique sur le bouton supprimer.

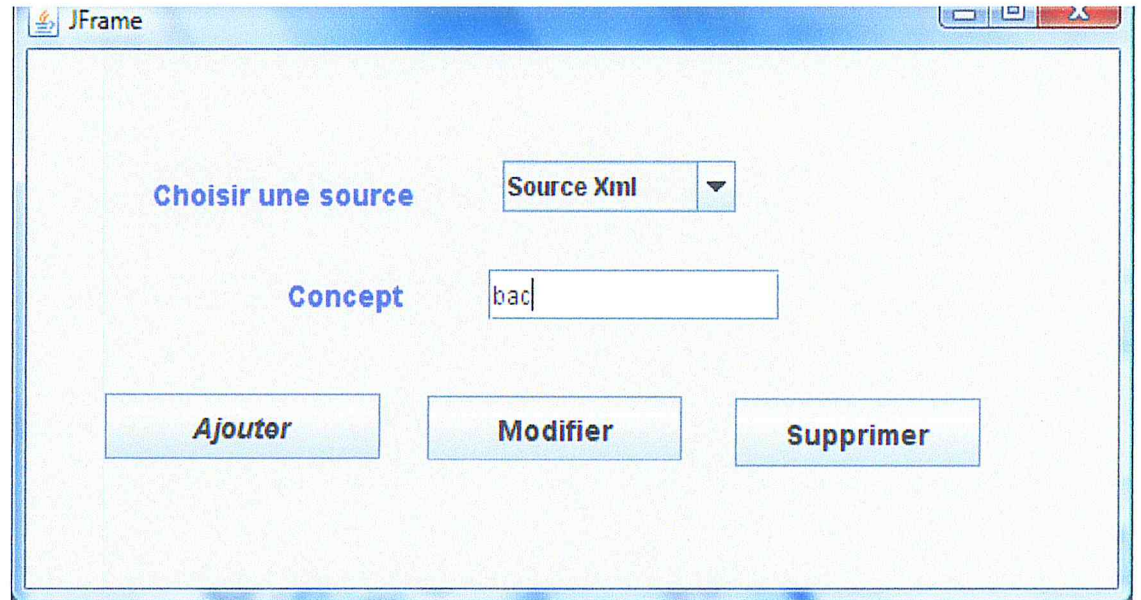


Figure IV.19: mise à jour des métadonnées.

### III. CONCLUSION

Dans ce chapitre, nous avons présenté les supports logiciels utilisés, ainsi que nos sources de métadonnées et les différents éditeurs qu'on a utilisé pour traiter ces sources, ensuite on a exposé l'environnement de développement choisi (langage de programmation, langage d'interrogation,.....etc.), dans le quel repose notre application.

Enfin, nous avons présenté la vue réelle de notre application, et des principaux modules qui la font fonctionner en toute simplicité, rapidité, grâce à une interface intuitive, simple respectant les concepts fondamentaux de l'IHM (interface homme machine).

**CONCLUSION ET  
PERSPECTIVE**

## CONCLUSION GENERALE :

Les travaux présentés dans ce mémoire se situent dans le contexte d'intégration des métadonnées, et plus particulièrement dans le cadre d'intégration par l'approche médiateur.

Le projet qui nous a été confié, porte sur la réalisation d'un système d'intégration de source de métadonnées par l'approche médiateur. Donc nous avons conçus une application qui répond aux objectifs fixés dans la phase de démarrage.

Dans le souci de présenter un travail regroupant un maximum d'éléments sur le concept de « métadonnée », nous avons présenté une synthèse sur les différentes typologies.

La démarche d'intégration des métadonnées au sein de notre domaine d'application, semble être l'étape la plus importante de cette étude du fait que les autres opérations décrites ne représente que les tâches de base d'un système d'intégration.

Afin d'élaborer ce projet, nous avons commencé le travail par la définition des besoins en faisant la collecte d'information nécessaire à notre étude préalable ensuite nous avons entamé l'étude conceptuelle suivant le modèle en cascade qui traite les différentes phases à savoir la conception détaillée et réalisation pour aboutir à la réalisation de l'application.

Notre projet de fin d'études nous a été d'un grand apport sur plusieurs plans principalement le plan conception, Nous avons appris l'aspect essentiel dans les applications d'intégration qui base sur les métadonnées. Sur le plan technique nous avons acquis une certaine expérience dans le domaine de développement par les technologies suivantes : Java, DOM et XQuery.

Ce projet est une base de réflexion concrète pour nous, qui peut être améliorée, corrigée ou complétée.

# Glossaires



## Glossaires

Signe	Désignation
API	Application Programming Interface
DC	Dublin Core
DCMI	Dublin Core Metadata Initiative
DOM	Document Object Model
DTD	Document Type Définition
GAV	Global As Views
HTML	Hyper Text Markup Language
JSP	Java Serveur Page
LAV	Local As Views
OMG	Object Management Group
OMT	Object Modeling Technique
OOSE	Object Oriented Software Engineering
OWL	Ontology Web Language
RDF	Resource Description Framework
RULEML	Rule Markup Language
SGML	Standard Generalized Markup Language)
SGBD	système de gestion de base de données
TEI	Text Encoding Initiative
UML	Unified Modeling Language
URI	Uniform Resource Identifiers
W3C	Consortium World Wide Web
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformations

# ANNEXES

## Annexe

Signe	Description
ACE ( Antigène Carcino-Embryonnaire)	L'antigène carcino embryonnaire est une protéine qui est présente chez le fœtus et qui augmente en cas de dédifférenciation cellulaire (en particulier lors de certains cancers).
Alpha FP(Alpha-fœtoprotéine)	une protéine qui n'est normalement produite que par le fœtus au cours de son développement.
ASLO (Antistreptolysines)	Les streptocoques $\beta$ hémolytiques du groupe A sécrètent une streptolysine O. Cet enzyme induit la formation d'anticorps
théophylline	La théophylline a un effet bronchodilatateur en relaxant le muscle lisse bronchique.
spermogramme	Un spermogramme est un examen médical au cours duquel on analyse le sperme d'un homme, généralement dans le cadre d'un bilan de stérilité d'un couple.
CPK MB	le dosage de la CPK MB permet de distinguer un infarctus d'une embolie pulmonaire
Insulinémie	C'est le taux d'insuline présente dans le sang.
Fibrinogène	Le fibrinogène est une protéine fabriquée par le foie qui est transformée en fibrine lors de la coagulation pour aboutir à la formation d'un caillot.
glycémie	le taux de sucre (glucose) dans le sang (Historiquement, ce terme est attribué à Claude Bernard). Sa mesure est exprimée en grammes par litre (g/l) ou en millimoles (mmol/l).
isoniazide	L'isoniazide est un antituberculeux indiqué en traitement curatif et préventif (chimio prophylaxie) de la tuberculose pulmonaire ou extrapulmonaire et des infections à mycobactéries atypiques en association avec d'autres antituberculeux.
Glycémie à jeun	Le glucose est le principal sucre de l'organisme. C'est lui qui apporte l'énergie à la plupart des cellules.

T4	La T4 représente 80% des hormones produites par la glande thyroïde
DARPA I3	DARPA I3 : Une architecture générique des solutions de médiation conçue par l'ARPA (Advanced Research Projects Agency) du département de la défense américaine <sup>1</sup> . Cette architecture baptisée I3 (Intelligent Integration of Information) est proposée par Gio Wiederhold.
L'anatomo-pathologie	ou anatomie pathologique <sup>1</sup> , est une spécialité médicale technique, humaine et vétérinaire, qui se consacre à l'étude des lésions macroscopiques et microscopiques des tissus pathologiques prélevés sur un sujet vivant ou décédé.
Les marqueurs tumoraux.	Les marqueurs tumoraux sont des molécules chimiquement définies ou non (connaissance ou non de leurs structures), synthétisées par les cellules tumorales, produites dans le tissu tumorale et sécrétées dans le sang ou dans les liquides biologiques
ECB	Examen simple permettant notamment de reconnaître les infections de la sphère respiratoire et de déterminer le germe en cause.

# **BIBLIOGRAPHIE**

## *REFERENCES BIBLIOGRAPHIQUE*

Signe	Titre d'ouvrage (Mémoire)	Auteur	Maison (organisme)	Année
[ANT, 2008]	Extraction et gestion des connaissances.	ANTIPOLIS Sophia		2008
[BAK, 2006]	Etude et proposition d'une architecture de médiation entre sources de données hétérogènes	BAKHTOUCHI Abdelghani	Institut national de formation en informatique « Oued Smar – Alger »	2006
[BAL, 2007]	Interopérabilité sémantique des systèmes d'informations distribuées	BALA Mahfoud	INI Alger	2007
[GAR, 2006]	« Médiation des données »	Georges Gardarin		2006
[DIO, 2007]	Spécification et en oeuvre d'un Formalisme de règles métier	DIOUF Mouhamed	Université BORDEAUX I	2007
[ELB, 2009]	ROMIE, une approche d'alignement d'ontologies à base d'instances	ELBYED Abdeltif	Institut national des télécommunications  CATIONS	2009
[FOR, 2009]	RDF et description RDF/XML	FORTIN Julie	Organisation de l'information numérique	2009
[GAY, 2006]	Une Architecture à Base d'Ontologies  pour la Gestion Unifiée des Données Structurées et non Structurées	GAYO Diallo	Université Joseph Fourier	2006

*these de doctorat*

[GIL, 2000]	Introduction to Metadata	Anne J. Gilliland-Swetland		2000
[HAC]	Les langages du Web sémantique	Mohand Saïd-Hacid	Université Claude Bernard Lyon	
[MEN, 2005]	Consolidation d'un modèle conceptuel de données de Master Data Management	Ludovic MENET	Thèse de Master à l'université de Marne-La-Vallée	2005-2006
[SAË, 2007]	Intégration sémantique de Données guidée par une Ontologie	SAËIS Fatiha	Thèse de doctorat à l'université de Paris-Sud	2007
[ZEN, 2009]	Conception et réalisation d'une application web pour la gestion du prêt entre bibliothèques	ZENDAOUI FAIROUZ et DRICI CHERIFA		2009-2010

## REFERENCES WEBGRAPHIQUES

Signe	Titre d'ouvrage (Mémoire)	Auteur	Site	Année
[ALT, 2011]			<a href="http://www.altova.com/semanticworks.html">http://www.altova.com/semanticworks.html</a>	2011
[CHA, 2004]	Cours de XML - Initiation aux Schema XML	G. Chagnon	<a href="http://www.gchagnon.fr/cours/xml/schema.html">http://www.gchagnon.fr/cours/xml/schema.html</a>	2004
[CHA, 2011]	Web Sémantique RDF	Charles Népote	<a href="http://websemantique.org/RDF">http://websemantique.org/RDF</a>	2011
[CHI, 2010]	Métadonnées et Dublin Core	Christophe Jacquet	<a href="http://openweb.eu.org/articles/dublin_core/">http://openweb.eu.org/articles/dublin_core/</a>	2010
[DIA ,2001]	Guide d'utilisation du Dublin Core	Diane Hillmann	<a href="http://www.bibl.ulaval.ca/DublinCore/usageguide-20000716fr.htm">http://www.bibl.ulaval.ca/DublinCore/usageguide-20000716fr.htm</a>	2001
[GRI]	« Conception d'un entrepôt de données (Data Warehouse) »	Yazid Grim	Developez.com	
[HAR and all, 2005]	RuleML Tutorial	Harold Boley , Benjamin Grososf, Said Tabet	<a href="http://www.ruleml.org/papers/tutorial-ruleml-20050513.html">http://www.ruleml.org/papers/tutorial-ruleml-20050513.html</a>	2005
[JAV, 2011]			<a href="http://www.futura-sciences.com/fr/definition//t/internet-2/d/java_485">http://www.futura-sciences.com/fr/definition//t/internet-2/d/java_485</a>	2011
[LAU, 2006]	UML 2 - de l'apprentissage à la pratique	Laurent Audibert	FNAC, amazon.fr	2006
[LIN]			<a href="http://www.springerlink.com/content/pq376517338/47848">http://www.springerlink.com/content/pq376517338/47848</a>	



[MQL, 2011]			<a href="http://www.futura-sciences.com/fr/definition/t/internet-2/d/mysql_4640">http://www.futura-sciences.com/fr/definition/t/internet-2/d/mysql_4640</a>	2011
[PHI, 2004]	Introduction à RDF	Philippe Lahaye	<a href="http://xmlfr.org/documentations/tutoriels/041015-0001">http://xmlfr.org/documentations/tutoriels/041015-0001</a>	2004
[STY, 2011]			<a href="http://www.stylusstudio.com/xml_feature_overview.html">http://www.stylusstudio.com/xml_feature_overview.html</a>	2011